Contamination Detection Is Not Collapse Monitoring: A Reproducible Study of AI-Text Detection and Model Collapse Across Eight Generators

Shaheer Saud, *Independent*, shaheersaud2004@gmail.com

Abstract

As language models increasingly train on their own output, two questions are routinely conflated: whether an individual *document* is machine-generated, and whether a *corpus* is undergoing *model collapse*, the progressive degradation of a model recursively trained on its own output [Shumailov2023], popularly nicknamed "Habsburg AI." **Our central claim is that synthetic-text detection and model-collapse monitoring are fundamentally different objectives: a detector can correctly flag synthetic documents while completely missing distributional degeneration, so treating detection as a proxy for collapse monitoring produces false confidence.** We make the case with deliberately cheap, interpretable tools: six pure-standard-library text statistics (the *SyntheticTextProbe*) and a single perplexity feature folded into a leave-one-model-out (LOMO) logistic-regression classifier. We first establish that lightweight detection *does* generalize to unseen generators: across eight LLMs on RAID [Dugan2024], the classifier reaches a mean held-out AUC of 0.915 ± 0.006 (10 seeds; 95% CI [0.911, 0.920]; 0.924 at our reference seed), versus 0.903 ± 0.007 for perplexity alone and 0.732 for the zero-shot heuristic, and a paired permutation test confirms the classifier's edge over perplexity alone is statistically significant (mean gain +0.013, $p = 0.002$, positive on 10/10 seeds). Swapping the bigram perplexity for a real GPT-2 (124M) signal does *not* help on average (0.895), a negative result we report with its in-distribution caveat; and an established GLTR-style likelihood detector outperforms our cheap heuristics zero-shot (0.995 vs. 0.915 on HC3), but that stronger detector is itself a per-document method and equally blind to collapse, which is the point. We then instrument recursive collapse directly with a trigram model, a fine-tuned distilGPT-2, and a modern Qwen2.5-0.5B: distribution-level metrics (vocabulary, n-gram diversity, and the generated text's reference perplexity, which crashes 69 → 7) track collapse cleanly and monotonically, while the very detector that flags individual documents stays flat (trigram) or rises but never crosses its threshold (neural, leaving ~91% of collapsed documents unflagged). The contribution is therefore not a new detector but a separation of concerns: **per-document contamination detection is not a reliable collapse monitor**, and a deployable collapse tripwire should track distribution-level diversity rather than aggregate per-document verdicts.

1. Introduction

The phrase "Habsburg AI" has become shorthand for a worry that is no longer hypothetical: as generative models produce an ever-larger share of online text, the corpora used to train the next generation of models are increasingly contaminated with machine output. Trained recursively on such data, models lose the tails of their distribution and drift toward bland, repetitive, low-diversity text, a phenomenon formalized as *model collapse* by [Shumailov2023], who showed that recursive training on generated data makes models forget. This raises a pair of intertwined questions that motivate this paper. First, can we *detect* synthetic text at the level of an individual document, cheaply and interpretably, in a way that generalizes beyond the specific generators we happened to train on? Second, even if we can flag individual documents, does per-document detection actually tell us what we want to know about *collapse*, a property of a distribution, not of any single text?

These two questions are often conflated. A detector that flags a paragraph as AI-written and a monitor that warns a corpus is collapsing sound like the same tool, but they are not, and treating them as interchangeable leads to false confidence. Put concretely: *a detector can correctly identify synthetic documents while completely failing to detect distributional degeneration.* That single fact is the heart of this paper. Much recent work pushes on detection (Section 2); comparatively little connects it back to the collapse phenomenon it is implicitly meant to guard against.

Our approach is deliberately minimal. Rather than fine-tune a large classifier, we ask how far one can get with six interpretable, hand-computed text statistics (repetition, filler density, lexical diversity, n-gram diversity, burstiness, and duplication against a reference corpus) implemented in pure Python with no dependencies. We combine these into a transparent heuristic scorer, the *SyntheticTextProbe*, and compare it against simple

baselines: a bigram-perplexity proxy and a real zero-shot GPT-2 (124M) signal. A single perplexity feature is then folded into a small logistic-regression classifier trained leave-one-model-out, so that every evaluation is on a generator the classifier has never seen. We use two real corpora: HC3 [Guo2023], a single-model (ChatGPT) human-versus-machine benchmark, and RAID [Dugan2024], which spans eight generators. We also instrument recursive collapse directly, retraining a trigram language model and a distilGPT-2 on their own output across successive generations while tracking diversity, perplexity, and detector score together.

Our findings form a clean separation of concerns: the cheap statistics plus perplexity *do* generalize to unseen generators (LOMO mean AUC 0.915), yet when we watch collapse happen, distribution-level signals track it cleanly and monotonically while the per-document detector stays flat or sub-threshold. We did not discover model collapse [Shumailov2023]; our contribution is to characterize *when* cheap detection generalizes, *when* it fails, and *why* per-document detection should not be mistaken for collapse monitoring.

Contributions

- **Our central claim: per-document detection is not a reliable collapse monitor.** Retraining a trigram LM, a distilGPT-2, and a modern Qwen2.5-0.5B on their own output, we show that distribution-level metrics catch collapse cleanly and monotonically, while the *same* per-document detector that flags AI text stays flat (trigram) or rises but never crosses its threshold (neural, ~91% of collapsed documents unflagged). A deployable collapse tripwire should therefore track distribution-level diversity, not aggregate document verdicts.
- **Statistically grounded evidence that cheap statistics plus perplexity generalize to unseen generators.** Under strict leave-one-model-out evaluation on RAID, the classifier reaches a mean held-out AUC of 0.915 ± 0.006 over 10 seeds (95% CI [0.911, 0.920]; 0.924 at the reference seed), significantly above perplexity alone (0.903 ± 0.007; non-overlapping CIs; paired permutation $p = 0.002$, 10/10 seeds) and far above the zero-shot heuristic (0.732 ± 0.005), with perplexity as the workhorse feature.
- **A characterization of *uneven* zero-shot transfer.** A single fixed heuristic detector transfers well to RLHF-tuned chat models but falls to near chance on GPT-2 base completions, mapping where cheap detection succeeds and fails.
- **A negative result on perplexity model size.** Replacing the bigram-perplexity feature with a real zero-shot GPT-2 (124M) signal does *not* improve mean LOMO AUC (0.895 vs. 0.924 at the reference seed); the in-distribution bigram and the zero-shot GPT-2 operate under different information conditions.
- **A lightweight, interpretable detector and LOMO classifier (the vehicle, not the headline).** A six-signal pure-standard-library scorer (the *SyntheticTextProbe*) plus a hand-rolled logistic-regression classifier, both designed for zero-install reproducibility, the instruments that make the claim above measurable.
- **A head-to-head with an established detector.** GLTR-style likelihood detection under GPT-2 outperforms our heuristics in the zero-shot setting (0.995 vs. 0.915 on HC3; 0.862 vs. 0.724 on pooled RAID), yet the same per-document limitation applies to the stronger detector, which reinforces the central claim.
- **A fully reproducible, near-zero-dependency artifact.** All code, fixed-seed splits, multi-seed runs, and result JSONs are released; the core runs on the Python standard library alone, with a single isolated environment only for the GPT-2 perplexity and neural-collapse experiments.

---

2. Related Work

Model collapse. The phenomenon we monitor in this work was identified by Shumailov et al. [Shumailov2023], who showed that language models trained recursively on their own generated outputs progressively forget the tails of the original data distribution, a degenerative process they term *model collapse* (and which, in its visible vocabulary-narrowing form, has been popularized as "Habsburg AI"). Their analysis establishes that successive generations lose rare events, contract their effective vocabulary, and drift toward low-variance text. We do not re-discover this effect; rather, we instrument it with cheap distribution-level statistics (vocabulary size, distinct-n-gram diversity, reference perplexity) and ask a distinct question: whether a *per-document* synthetic-text detector can flag the individual outputs of a collapsing model. As our results show, these are not the same problem.

Statistical and zero-shot AI-text detection. A parallel line of work detects machine-generated text from surface and likelihood statistics rather than a trained discriminator. GLTR [Gehrmann2019] visualizes per-token model probabilities to expose the unnaturally high-likelihood tokens characteristic of sampled text, and DetectGPT [Mitchell2023] formalizes a zero-shot test based on the observation that machine text tends to occupy regions of negative log-probability curvature under the generating model. Deployed commercial detectors such as GPTZero [Tian2023] make per-

document synthetic-vs-human judgments at scale. Our SyntheticTextProbe sits in this tradition but is deliberately lighter: it combines six interpretable text statistics (repetition, filler density, lexical and n-gram diversity, burstiness, and reference-corpus duplication) with a small perplexity signal, using only the Python standard library, trading the access to a generating model's token probabilities that DetectGPT and GLTR assume for full transparency and zero-install reproducibility. We compare directly against the GLTR statistics computed under GPT-2 in Section 5.7, and find, unsurprisingly, that the white-box likelihood detector outperforms our heuristics in the zero-shot setting; our point is not that the heuristics are stronger, but that the per-document/distribution distinction holds regardless of detector strength.

Reliability limits. A cautionary body of work argues that AI-text detection is fundamentally fragile. Sadasivan et al. [Sadasivan2023] show that paraphrasing and other simple transformations can drive detector accuracy toward chance, and argue on theoretical grounds that as generators approach the human text distribution, reliable detection becomes impossible. Our findings are consistent with this caution rather than in tension with it: our zero-shot heuristic transfers unevenly across generators (strong on RLHF-tuned chat models, near chance on GPT-2 base completions), and even the trained classifier's generalization to unseen generators leans heavily on a single perplexity feature. We therefore frame our positive results as characterizations of *when* cheap statistics work, not as a claim that per-document detection is robust in the adversarial limit.

Datasets and benchmarks. We evaluate on two public resources. HC3 [Guo2023] pairs real human answers with ChatGPT answers to the same questions, providing a controlled single-generator comparison. RAID [Dugan2024] is a large shared benchmark of machine-generated text spanning many generators (here gpt2, gpt3, gpt4, chatgpt, llama-chat, mistral-chat, cohere, and mpt-chat) together with human text and model/attack labels, enabling cross-model and leave-one-model-out evaluation. Using HC3 for a single-model snapshot and RAID for multi-generator generalization lets us separate the easy in-distribution case from the harder question of transfer to unseen generators.

---

3. Method

Our method comprises three components: (i) **SyntheticTextProbe**, a heuristic per-document scorer built entirely from cheap, interpretable text statistics; (ii) a set of **baselines** that bracket its difficulty, ranging from a trivial sanity floor to a real neural language model; and (iii) a small **leave-one-model-out (LOMO) logistic-regression classifier** that combines the detector's signals with a single perplexity feature to test whether these signals generalize to unseen generators. All code is pure Python standard library except for the GPT-2 perplexity feature, which uses a single isolated PyTorch/Transformers environment.

3.1 The SyntheticTextProbe

The detector is a deliberately simple instrument, not the paper's contribution; we name it the *SyntheticTextProbe* to keep its function, scoring a single document, distinct from the model-collapse phenomenon ("Habsburg AI") that the study is about. The SyntheticTextProbe maps a document to a synthetic-likelihood score in $[0, 100]$, where higher means more likely to be machine-generated. The score is a fixed-weight linear combination of six signals, each computed from the raw text and normalized to $[0, 1]$. The design goal is interpretability and zero-install reproducibility: every signal is a simple statistic over tokens, sentences, and n-grams.

Three of the six signals, lexical diversity, n-gram diversity, and burstiness, are *human-high*: human text tends to score higher on them. To keep the convention that a larger contribution always means *more* synthetic, these three signals are **inverted** (we use $1 - s$) before weighting. The remaining three, repetition, filler density, and duplicate similarity, are *synthetic-high* and used directly. Table 1 lists each signal, its weight, and its orientation.

Table 1. The six signals of the SyntheticTextProbe, their fixed weights (which sum to 1.0), what each measures, and whether it is used directly or inverted. Inverted signals are *human-high*: human text scores higher, so we use $1 - s$ so that every weighted contribution increases with synthetic likelihood.

The six signals are defined as follows.

- `repetition` (weight 0.18, direct). The fraction of token trigrams in the document that are repeats, i.e. the share of trigram occurrences belonging to trigrams that appear more than once. Machine text that loops or paraphrases itself elevates this value.
- `filler_density` (weight 0.20, direct). The number of canned filler phrases (a fixed phrase list, e.g. hedging and boilerplate connectives) found in the document, normalized to a per-~50-token rate. This targets the formulaic connective tissue characteristic of instruction-tuned output.
- `lexical_diversity` (weight 0.16, inverted). The type-token ratio: the number of distinct word types divided by the total number of word tokens. Human writing tends to use a richer vocabulary,

Table 1: The six signals of SyntheticTextProbe: weight, what each measures, and orientation.

Signal	Weight	What it measures	Orientation
`repetition`	0.18	Fraction of trigrams that are repeats (repeated-trigram fraction)	synthetic-high (direct)
`filler_density`	0.20	Count of canned filler phrases per ~50 tokens	synthetic-high (direct)
`lexical_diversity`	0.16	Type-token ratio (distinct words / total words)	human-high (**inverted**)
`ngram_diversity`	0.16	Distinct-trigram ratio (distinct trigrams / total trigrams)	human-high (**inverted**)
`burstiness`	0.15	Coefficient of variation of sentence lengths	human-high (**inverted**)
`duplicate_similarity`	0.15	Maximum Jaccard overlap against a reference corpus of known model outputs	synthetic-high (direct)

so this signal is *human-high* and inverted before weighting.

- **`ngram_diversity`** (weight 0.16, inverted). The distinct-trigram ratio: the number of distinct trigrams divided by the total number of trigrams. Like lexical diversity, this is *human-high* and inverted.
- **`burstiness`** (weight 0.15, inverted). The coefficient of variation (standard deviation divided by mean) of sentence lengths in tokens. Human prose mixes short and long sentences, producing higher burstiness; machine text is often more uniform. This signal is *human-high* and inverted.
- **`duplicate_similarity`** (weight 0.15, direct). The maximum Jaccard similarity between the document's token set and that of any document in a reference corpus of known model outputs. When no reference corpus is supplied, this signal is defined to be 0 and contributes nothing.

Each normalized (and, where applicable, inverted) signal value $s_i \in [0, 1]$ is combined with its fixed weight w_i, and the result is scaled to a 0–100 score:

$$\text{score} = 100 \cdot \sum_{i=1}^{6} w_i \, s_i, \qquad \sum_{i=1}^{6} w_i = 1.0.$$

A document is **predicted synthetic** when score $\geq$ 50, and predicted human otherwise. Because the weights sum to 1.0 and each signal lies in $[0, 1]$, the score is bounded in $[0, 100]$ by construction; no per-dataset tuning or learned threshold is used.

3.2 Baselines

We compare the detector against three baselines spanning a wide range of cost and modeling power.

- **PerplexityBaseline (bigram).** A word-level bigram language model with add-k smoothing ($k = 0.5$), fit on a reference corpus. Each document is scored by its mean per-token cross-entropy under this model, which serves as a perplexity proxy. The sign of the score is calibrated on the training split so that the positive (synthetic) class corresponds to the appropriate direction. This baseline is cheap and standard-library-only, but, when re-fit per fold, it has access to in-distribution text (see Section 4).
- **MeanWordLength (floor).** The average word length of the document, used as a trivial sanity-floor baseline. It captures essentially no signal beyond gross stylistic differences and exists to establish the lower bound a method must clear to be meaningful.
- **Real GPT-2 perplexity.** The perplexity of each document under the pretrained GPT-2 (124M) model [Radford2019], used fixed and zero-shot (no fitting). Perplexities are computed on an Apple MPS GPU and cached. Unlike the bigram baseline, GPT-2 is never fit to any in-distribution split, so it is the more deployable, but, as we report in Section 5, not uniformly stronger, perplexity signal.

3.3 The LOMO logistic-regression classifier

To test whether the cheap signals generalize *beyond the generators seen at training time*, we train a small logistic-regression classifier and evaluate it under a **leave-one-model-out (LOMO)** protocol: the classifier is trained on documents from seven generators (plus human text) and tested on the held-out eighth generator, which it has never seen. This isolates cross-generator transfer from in-distribution fitting.

The classifier is a hand-rolled logistic regression trained by gradient descent, with internal z-score standardization of each feature and L_2 regularization (strength 10^{-3}). It operates over **seven features**: the six SyntheticTextProbe signals (Section 3.1) plus **one perplexity feature**. We instantiate the perplexity feature two ways and report both: (i) the bigram-LM perplexity proxy of Section 3.2, re-fit in-distribution per fold; and (ii) the fixed, zero-shot real GPT-2 (124M) perplexity. Comparing the two isolates the contribution of a genuine neural language model as a feature, against an otherwise identical pipeline.

Because logistic regression standardizes its inputs, the magnitude of each learned coefficient (in standardized units) serves as a direct, comparable measure of feature importance; we report these in Section 5 to identify which signals carry the cross-generator signal and which are inert. The training and test splits, 2000 human training documents and 400 disjoint human test documents, balanced against the machine class, are held constant across all folds so that differences across held-out generators reflect the generators themselves, not

changes in the human sample. The fixed random seed (1234) governs all sampling, initialization, and shuffling.

---

4. Experimental Setup

We evaluate the detector, the baselines, and the leave-one-model-out (LOMO) classifier on two public corpora of paired human and machine text: HC3 [Guo2023] and RAID [Dugan2024]. Throughout, the positive class is synthetic (LLM-generated) text, the global random seed is 1234, and every text is truncated to roughly 250 words before feature extraction so that document length cannot leak label information.

4.1 Datasets

HC3 [Guo2023] pairs real human answers with real ChatGPT answers to the *same* questions, which controls for topic and prompt. We draw a balanced subset of 2500 human and 2500 ChatGPT documents, each truncated to ~250 words. The corpus is fetched over HTTP as a single JSONL file (~70 MB). Because HC3 contains exactly one machine generator (ChatGPT), it serves as a single-model, relatively easy snapshot rather than the headline cross-generator evaluation.

RAID [Dugan2024] contains machine text from many generators alongside human text, annotated with a model label and an attack label. We restrict to `attack=none` and sample 8 generators × 400 documents plus 3000 human documents, each truncated to ~250 words. The eight generators are **gpt2, gpt3, gpt4, chatgpt, llama-chat, mistral-chat, cohere, and mpt-chat**, spanning base-completion models (gpt2), older instruction-tuned models, and current RLHF chat models. RAID is fetched via the HuggingFace datasets-server REST API.

4.2 Splits

HC3 (70/30). The HC3 subset is divided into a 70/30 train/test split, yielding a held-out test set of 1500 documents (balanced human vs. ChatGPT). Result 1 (single-model detection) and the per-signal ablation (Result 2) are reported on this split.

Cross-model zero-shot (60/40). For the cross-model evaluation (Result 3), the detector is applied as a single, *fixed* zero-shot scorer and compared against human text per generator on RAID. The bigram perplexity baseline reported alongside it is re-fit on a RAID train split under a 60/40 split; this gives the perplexity baseline an in-distribution advantage, so it is *not* a fair zero-shot comparison (Section 5.3).

Leave-one-model-out (LOMO). For the generalization experiments (Results 4 and 5), we adopt a leave-one-model-out protocol over the eight RAID generators: the classifier is trained on 7 generators (plus human) and tested on the held-out 8th, rotating across all eight folds. The human partition is held constant across folds, **2000 human training documents and 400 human test documents, disjoint and balanced**, so that only the machine-generator composition changes between train and test. This isolates generalization to an *unseen* generator from any shift in the human distribution.

4.3 Metrics

We report **AUC** (area under the ROC curve) as the primary, threshold-free metric, computed with a rank-based (Mann–Whitney U) formula with the positive class fixed to synthetic/LLM text. We also report **accuracy, precision, recall, and F1** at the best-threshold operating point (the detector's default decision rule is a score threshold of 50 ⇒ predicted synthetic). For the recursive-collapse experiments (Result 6) we also track distribution-level quantities, vocabulary size, distinct-trigram diversity, type–token ratio, and reference-model perplexity, alongside the per-document detector score.

4.4 Reproducibility and Compute

The core pipeline, the six-signal detector, the bigram-perplexity and mean-word-length baselines, the LOMO logistic-regression classifier, the cross-model evaluation, and the trigram collapse experiment, is written in the **Python standard library only**, with zero third-party install required. Real corpora are pulled over HTTP at run time: HC3 as a ~70 MB JSONL download, and RAID through the HuggingFace datasets-server REST API. A single isolated virtual environment (torch 2.12 + transformers) is used *only* for the two neural components: the real GPT-2 (124M) perplexity feature (Result 5) and the neural distilGPT-2 collapse experiment (Result 6); both run on the Apple MPS GPU, and the GPT-2 perplexities are cached. The global seed 1234 is fixed everywhere, and all code (`detector.py`, `baselines.py`, `evaluate.py`, `crossmodel.py`, `lomo.py`, `gpt2_perplexity.py`, `collapse_experiment.py`, `collapse_gpt2.py`) and result JSON files are included in the repository.

---

5. Results

5.1 HC3 Single-Model Held-Out (Positive = ChatGPT)

We first evaluate on HC3 [Guo2023], where real human answers are paired against ChatGPT answers to the *same* questions, holding out a balanced test set of 1500 documents. This is a deliberately favorable setting: a single, relatively distinctive generator. Table 2 reports the held-out scores.

Table 2. HC3 held-out performance (positive class = ChatGPT). AUC computed via the rank-based (Mann-Whitney) formula.

The SyntheticTextProbe reaches AUC 0.912 (acc 0.845, F1 0.850), ahead of the re-fit bigram perplexity baseline (AUC 0.832) and well clear of the trivial mean-word-length floor (AUC 0.646). The floor baseline's high recall (0.940) at low precision (0.531) is the expected signature of a near-degenerate classifier that predicts "synthetic" for most documents; its AUC of 0.646 confirms it carries only weak signal. Figure 5 shows the corresponding ROC curves on HC3.

HC3 is a **single-model** snapshot: every positive document comes from ChatGPT, and the human/ChatGPT contrast on shared questions is relatively easy. This result is a calibration point, not the headline of the paper. The cross-model and leave-one-model-out experiments below are the more demanding tests of whether cheap text statistics generalize.

5.2 Per-Signal Ablation on HC3

To understand which of the six signals carry the detector, we ablate each one on HC3. The full detector scores AUC 0.912. For each signal we report the *leave-one-out* (LOO) AUC when that signal is removed, the corresponding **dAUC** (AUC lost relative to the full detector), and the **solo** AUC obtained using that signal alone. Table 3 sorts signals by dAUC.

Table 3. Per-signal ablation on HC3 (full detector AUC = 0.912). dAUC = AUC lost when the signal is removed; solo = AUC using only that signal.

Two signals do most of the work. Burstiness (sentence-length coefficient of variation, inverted) and lexical diversity (type-token ratio, inverted) each cost the largest amount when removed (dAUC 0.037 and 0.036) and are also the most informative as standalone features on HC3, though note that burstiness has only a modest solo AUC (0.735) yet the largest dAUC, indicating it contributes information that the other signals do not duplicate. Repetition adds a smaller but nontrivial increment (dAUC 0.012). The distinct-trigram ratio (ngram_diversity) and filler density are largely redundant here (dAUC 0.004 and 0.003); filler density in particular is weak in isolation (solo 0.613). Finally, duplicate_similarity is **inert on HC3** (dAUC 0.000, solo 0.500): with no reference corpus of known model outputs supplied in this setting, the signal is defined to be zero for every document and therefore carries no information.

5.3 Cross-Model Zero-Shot Detection (RAID)

The harder question is whether one *fixed* detector, calibrated once, transfers to generators it was never tuned against. We freeze the SyntheticTextProbe and score each RAID generator against human text (attack = none, 400 samples per generator, ~250 words each) [Dugan2024]. For reference we also report the bigram PerplexityBaseline; we stress that this baseline is **re-fit in-distribution** on a RAID train split and therefore is *not* a fair zero-shot comparison, it has seen the target distribution and enjoys a corresponding advantage. Table 4 reports per-generator AUC; Figure 6 plots the cross-model bars.

Table 4. Cross-model AUC, one fixed zero-shot SyntheticTextProbe vs. human, per generator (RAID). Bigram perplexity (re-fit in-distribution) shown for reference only, not a fair zero-shot comparison.

Across the eight generators the fixed SyntheticTextProbe averages AUC 0.710 (pooled 0.723), ranging from 0.566 on gpt2 to 0.829 on chatgpt.

The zero-shot heuristic transfers **unevenly**. It is strongest on RLHF / chat-tuned models, chatgpt (0.829), llama-chat (0.794), mistral-chat (0.764), gpt3 (0.759), whose canned, low-burstiness, low-diversity output matches exactly what the six signals were designed to catch. It degrades sharply on base-style completions: on gpt2 (0.566) it is close to chance, and cohere (0.610) and mpt-chat (0.654) are also weak. The bigram perplexity column is consistently higher (pooled 0.908), but this is *expected and not a fair contest*: that baseline was re-fit on in-distribution RAID data, whereas the SyntheticTextProbe is genuinely zero-shot. The takeaway is not that perplexity beats the heuristics, but that a single fixed heuristic detector does not transfer uniformly across generator families, motivating the trained leave-one-model-out classifier in Section 5.4.

5.4 Leave-One-Model-Out Generalization

The cross-model results in Section 5.3 measure how a *fixed, zero-shot* heuristic transfers to generators it never saw. We now ask the complementary question: if we are allowed to *train* a classifier, but never on the target generator, how well does it generalize to that unseen generator? This is the leave-one-model-out (LOMO) protocol described in Sections 3 and 4. For each of the eight RAID generators in turn, we hold that generator out entirely, train the seven-feature logistic regression (the six detector signals plus one perplexity feature) on the remaining seven generators plus human text, and evaluate on the held-out generator versus held-out human text. The perplexity feature in this section is the add-k bigram language model, re-fit on the in-distribution reference split within each fold.

Table 2: HC3 single-model held-out detection (positive class: ChatGPT).

Method	AUC	Acc	Prec	Rec	F1
SyntheticTextProbe	0.912	0.845	0.824	0.879	0.850
PerplexityBaseline (bigram)	0.832	0.765	0.748	0.799	0.772
MeanWordLength (floor)	0.646	0.555	0.531	0.940	0.679

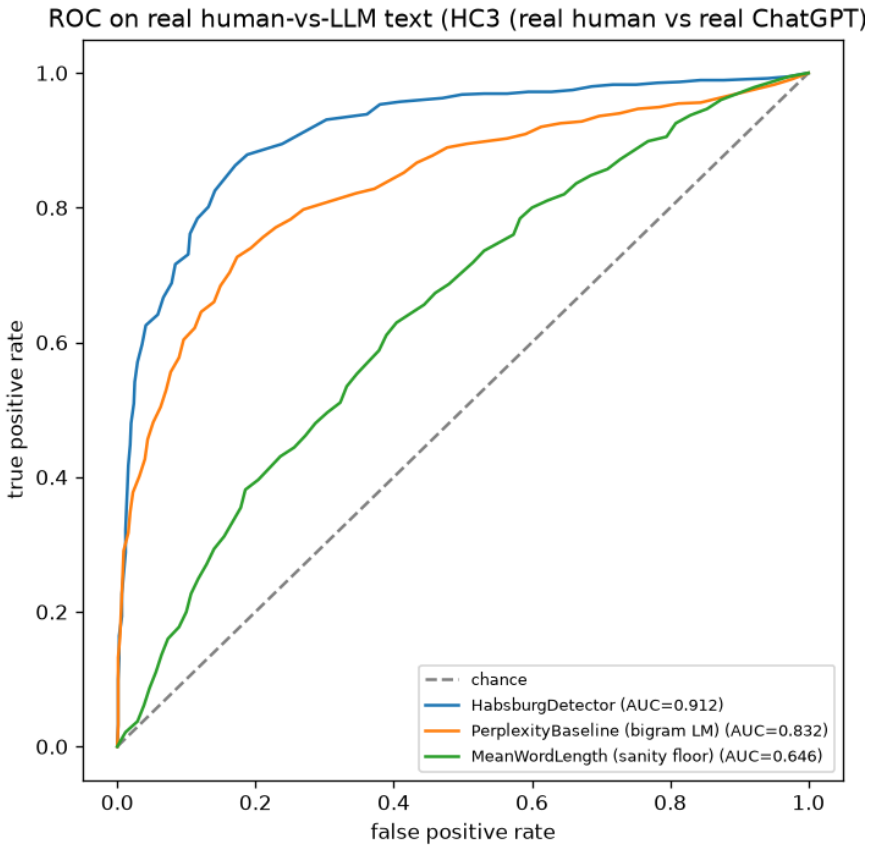


Figure 5: ROC on HC3 (real human vs. ChatGPT): SyntheticTextProbe against the perplexity and word-length baselines.

We compare three systems per fold: the trained LOMO **classifier**; **perplexity-alone**, i.e. the same bigram perplexity feature used as a single-feature scorer; and the **zero-shot** SyntheticTextProbe from Section 5.3, which is never trained and is shown here as the untrained reference. Results are in Table 5 and Figure 7.

Table 5. Leave-one-model-out AUC (RAID). Each row holds out one generator; the classifier and perplexity-alone are trained on the other seven generators plus human; zero-shot is the untrained SyntheticTextProbe. Positive class = synthetic.

Table 5 reports our reference-seed run (seed 1234); the 10-seed mean is 0.915 ± 0.006 (Section 5.4.1). The trained classifier generalizes to unseen generators with a mean AUC of 0.924 at this seed (0.915 across seeds), a large improvement over the untrained zero-shot heuristic (0.732). The gain comes almost entirely from being allowed to fit *any* in-distribution data at all, not from the target generator specifically: even with the target generator excluded, the seven-generator training mix plus human text is enough to calibrate the feature weights for the held-out model. The classifier holds up even on the two hardest generators, gpt2 (0.834) and cohere (0.837), where the zero-shot heuristic was near chance or weak (0.594 and 0.636), confirming that the generalization is real and not an artifact of the easy RLHF-chat generators.

The lift of the trained classifier over perplexity-alone is real but **modest**: +0.013 in mean AUC (0.924 vs 0.911). On most folds the two are within a few thousandths of each other, and on three folds (gpt2, gpt3, cohere) perplexity-alone is marginally *ahead* of the full classifier. The classifier's advantage is concentrated on a few generators: mpt-chat (+0.065, 0.950 vs 0.885) and gpt4 (+0.039, 0.907 vs 0.868). In other words, the cheap text statistics earn their place mainly where the perplex-

Table 3: Per-signal ablation on HC3 (full-model AUC 0.912). dAUC is the AUC lost when the signal is removed; Solo is the AUC using that signal alone.

Signal	LOO AUC	dAUC	Solo AUC
burstiness	0.874	0.037	0.735
lexical_diversity	0.876	0.036	0.848
repetition	0.899	0.012	0.848
ngram_diversity	0.908	0.004	0.850
filler_density	0.909	0.003	0.613
duplicate_similarity	0.912	0.000	0.500

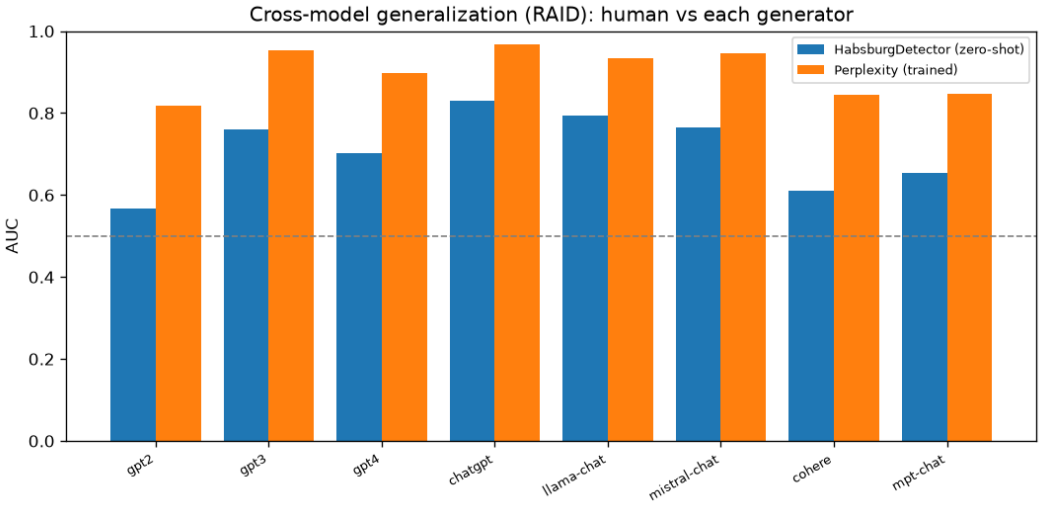


Figure 6: Cross-model detection on RAID: one fixed zero-shot detector evaluated against each of eight generators.

ity signal alone is comparatively weak; where perplexity already separates the classes cleanly, they add little. As Section 5.4.1 shows, this lift is small but consistent and statistically significant across seeds.

5.4.1 Significance across seeds Because the absolute AUC depends on which human documents land in the train and test splits, we repeat the entire leave-one-model-out experiment over **10 random seeds** (each re-draws the human partition; the reference seed 1234 is included). The classifier's mean held-out AUC across seeds is 0.915 ± 0.006 (95% CI [0.911, 0.920]), perplexity-alone is 0.903 ± 0.007 (95% CI [0.898, 0.908]), and the zero-shot heuristic is 0.732 ± 0.005 (95% CI [0.729, 0.735]); the classifier and perplexity-alone confidence intervals do not overlap. A paired sign-flip permutation test on the ten per-seed differences gives a mean gain of +0.013 ± 0.002, positive on **all 10 of 10 seeds**, with $p = 0.002$, so the classifier's advantage over perplexity-alone, while modest, is statistically significant and robust to resampling. The per-seed absolute AUC ranges from 0.903 to 0.924, which is why we report the 10-seed mean, rather than any single run, as the headline. (We multi-seed the headline bigram-feature LOMO result here; the GPT-2-feature variant of Section 5.5 and the collapse experiments of Section 5.6 remain single-seed and are read as point estimates.)

This is made explicit by the feature-importance analysis. Table 6 reports the mean absolute standardized coefficient of each feature across the eight LOMO folds (the logistic regression standardizes features internally via z-scoring, so these coefficients are directly comparable in magnitude).

Table 6. LOMO classifier feature importance: mean absolute standardized logistic-regression coefficient across the eight folds (bigram-perplexity feature).

Perplexity is the workhorse feature by a wide margin (2.73), more than twice the next-largest contributor. Among the cheap statistics, lexical diversity (1.24) and repetition (0.58) carry most of the remaining signal, consistent with the HC3 ablation in Section 5.2 where these were also among the most informative individual signals. Duplicate_similarity is exactly 0.00 because no reference corpus is supplied in this experiment, so it is inert, as it was on HC3. The picture is consistent: perplexity does the heavy lifting, a small number of interpretable statistics provide a modest but real top-up, and that top-up is largest on the generators where perplexity is weakest.

Table 4: Cross-model zero-shot detection on RAID, per generator. Perplexity is re-fit in-distribution and shown for reference only.

Generator	Habsburg (zero-shot)	Perplexity (in-distribution ref.)
gpt2	0.566	0.819
gpt3	0.759	0.953
gpt4	0.703	0.897
chatgpt	0.829	0.968
llama-chat	0.794	0.934
mistral-chat	0.764	0.945
cohere	0.610	0.845
mpt-chat	0.654	0.847
Pooled (all)	0.723	0.908

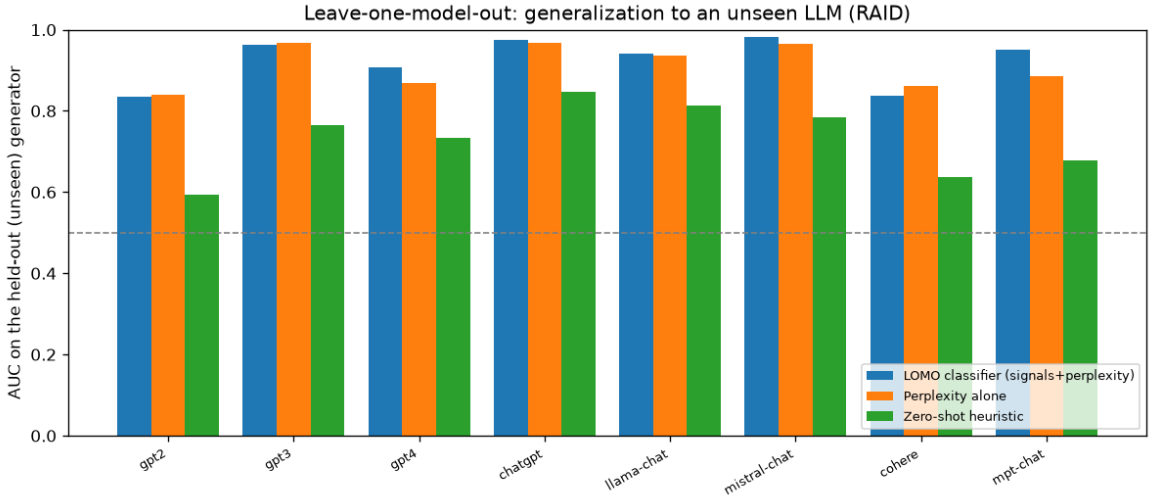


Figure 7: Leave-one-model-out: AUC on the held-out (unseen) generator for the trained classifier, perplexity alone, and the zero-shot heuristic.

5.5 Real GPT-2 Perplexity versus the Bigram Language Model

A natural hypothesis is that the modest, perplexity-dominated classifier of Section 5.4 should improve if we replace the toy add-k bigram language model with a real neural language model. We test this directly by re-running the identical LOMO protocol with one change: the single perplexity feature is now the perplexity of each text under the pretrained GPT-2 (124M) model, used fixed and zero-shot (no per-fold re-fitting), computed on Apple MPS and cached. Everything else, the six cheap signals, the logistic regression, the splits, the seed, is held constant. Results are in Table 7.

Table 7. Leave-one-model-out AUC with the GPT-2 perplexity feature (RAID). Same protocol as Table 5, with the bigram perplexity feature replaced by real GPT-2 (124M) perplexity, fixed and zero-shot.

The result is a **surprising negative one**: the real GPT-2 perplexity feature did *not* beat the toy bigram language model on average (both compared here at the reference seed; the GPT-2 variant was not multi-seeded). The GPT-2-feature classifier reaches a mean AUC of 0.895, below the 0.924 of the bigram-feature classifier in Section 5.4; as a standalone scorer, GPT-2 perplexity-alone (0.864) likewise trails the bigram perplexity-alone (0.911). The neural model helped only on a minority of folds, most clearly gpt2 (perplexity-alone 0.857 vs 0.840) and llama-chat (0.964 vs 0.936), and it actively *hurt* on others, especially gpt4 (0.756 vs 0.868), mpt-chat (0.761 vs 0.885), and cohere (0.778 vs 0.862).

This comparison is **not apples-to-apples**. The bigram language model is re-fit *in-distribution* on a RAID train split within each fold, so it adapts to the domain and the surrounding generator mix; GPT-2 perplexity is fixed and fully zero-shot, with no exposure to the RAID distribution at all. The bigram model therefore enjoys a fitting advantage that the frozen GPT-2 does not. Read in that light, the more deployable interpretation is the charitable one: a fixed, zero-shot neural perplexity reaches mean AUC ~0.86–0.90 *with no corpus fitting whatsoever*, which is arguably the more transferable signal even though it loses the head-to-head against

Table 5: Leave-one-model-out AUC on the unseen generator, bigram-perplexity feature.

Held-out generator	Classifier	Perplexity-alone	Zero-shot
gpt2	0.834	0.840	0.594
gpt3	0.962	0.967	0.765
gpt4	0.907	0.868	0.734
chatgpt	0.976	0.968	0.848
llama-chat	0.942	0.936	0.814
mistral-chat	0.981	0.965	0.784
cohere	0.837	0.862	0.636
mpt-chat	0.950	0.885	0.679
Mean	0.924	0.911	0.732

Table 6: LOMO classifier feature importance (mean standardized coefficient), bigram-perplexity feature.

Feature	Mean |standardized coef|
perplexity	2.73
lexical_diversity	1.24
repetition	0.58
burstiness	0.29
filler_density	0.14
ngram_diversity	0.13
duplicate_similarity	0.00

an in-distribution-tuned bigram. The headline lesson is not that GPT-2 is bad, but that **a bigger language model is not automatically a better detection feature**, feature quality depends on the match between the scoring model and the target distribution, not on raw model capacity.

The feature-importance shift under the weaker perplexity signal is itself informative. Table 8 reports the GPT-2-feature classifier's mean absolute standardized coefficients.

Table 8. LOMO feature importance with the GPT-2 perplexity feature: mean absolute standardized coefficient across the eight folds. Compare to Table 6.

When the perplexity feature is the weaker GPT-2 signal, the classifier compensates by leaning *more heavily* on the cheap statistics: lexical_diversity rises from 1.24 to 1.46, repetition from 0.58 to 0.90, burstiness from 0.29 to 0.61, and both ngram_diversity and filler_density roughly triple. The perplexity coefficient itself drops (2.73 to 2.49). This is exactly the behavior one would hope for from an interpretable additive model: when the dominant feature degrades, the lightweight signals take up the slack rather than the model failing outright. It also reinforces the central message of this paper, the cheap text statistics are most valuable when the expensive signal is weakest, and a classifier that combines both degrades gracefully rather than catastrophically. (As before, duplicate_similarity remains 0.00, inert without a reference corpus.)

5.6 Model Collapse

Having established that the detector and classifier generalize across generators for *per-document* detection (Sections 5.3–5.5), we now turn to a structurally different question: when a generative model is recursively retrained on its own output and undergoes *model collapse* [Shumailov2023], do our signals see it coming? We instrument two collapse processes, one symbolic, one neural, and track both distribution-level diversity metrics and the per-document detector score across generations. The seed corpus is 150 documents drawn from the RAID human pool.

We did not discover model collapse; [Shumailov2023] did. Our contribution here is narrow and diagnostic: we measure *which* of our signals respond to collapse, and we contrast distribution-level monitoring with per-document detection.

Trigram language model. We retrain a trigram language model on its own sampled output for six generations. The distributional symptoms of collapse are unambiguous: the vocabulary shrinks from 6707 to 1287 distinct tokens (-81%), trigram diversity falls from 0.943 to 0.270, and the type-token ratio drops from 0.212 to 0.113. The per-document detector, however, is effectively blind: its mean score *falls* from 16.7 to 11.9, it does not rise toward the synthetic threshold at all. Estimated contamination (fraction of generated documents the detector flags as synthetic) creeps only from 0% to 1%. A monotonically collapsing distribution leaves the detector flat.

Table 7: Leave-one-model-out AUC on the unseen generator, real GPT-2 perplexity feature.

Held-out generator	Classifier (GPT-2 feat.)	Perplexity-alone (GPT-2)	Zero-shot
gpt2	0.811	0.857	0.594
gpt3	0.920	0.934	0.765
gpt4	0.895	0.756	0.734
chatgpt	0.974	0.944	0.848
llama-chat	0.948	0.964	0.814
mistral-chat	0.976	0.916	0.784
cohere	0.725	0.778	0.636
mpt-chat	0.909	0.761	0.679
Mean	0.895	0.864	0.732

Table 8: LOMO feature importance with the GPT-2 perplexity feature.

Feature	Mean |standardized coef| (GPT-2)	Cf. bigram (Table 6)
perplexity	2.49	2.73
lexical_diversity	1.46	1.24
repetition	0.90	0.58
burstiness	0.61	0.29
ngram_diversity	0.39	0.13
filler_density	0.38	0.14
duplicate_similarity	0.00	0.00

Neural model (distilGPT-2). We repeat the experiment with a neural generator: a fresh distilGPT-2 is fine-tuned for 3 epochs on the previous generation's output (Apple MPS), for five generations. The distributional symptoms again appear cleanly: vocabulary shrinks from 6707 to 1012 (-85%) and trigram diversity falls from 0.943 to 0.639. This time the detector *does* respond, its mean score rises from 16.7 to 27.1, but it never crosses the threshold of 50, so estimated contamination reaches only 9%; roughly 91% of collapsed documents remain unflagged. The cleanest and most monotonic signal is reference perplexity: the perplexity of the generated text under a *fixed* reference distilGPT-2 collapses from 69.2 to 7.4, decreasing at every generation as the text becomes ever more predictable to the model that spawned its lineage.

Table 9 reports the per-generation trajectory for both experiments. The contrast is the point: every distribution-level metric (vocabulary, trigram diversity, reference perplexity) moves monotonically and decisively, while the per-document detector either stays flat (trigram) or rises but stays sub-threshold (neural).

Table 9. Per-generation collapse trajectory. Trigram LM (six generations, top) and neural distilGPT-2 (five generations, bottom). Detector score is the mean SyntheticTextProbe synthetic-likelihood (threshold 50). Reference perplexity (neural only) is under a fixed distilGPT-2.

Figure 8 visualizes the neural trajectory: the left panel overlays trigram diversity and detector score across generations (diversity falling, detector rising but capped well below threshold), and the right panel shows reference perplexity collapsing from 69 to 7.

Does this survive scale and a second corpus? The runs above use 150-document corpora; a fair objection is that small samples collapse trivially. We therefore re-ran the trigram collapse while varying the per-generation corpus size over {150, 1{,}500, 15{,}000} and the seed corpus over two domains, RAID human text and HC3 human answers, holding generation 0 fixed (Figure 10, *figure10_collapse_scaling.png*). Two results hold across all six configurations. First, larger corpora collapse more *slowly*: a bigger sample keeps the distributional tail alive longer, so vocabulary loss after five generations falls from -96% at 150 documents to -32% (RAID) and -17% (HC3) at 15,000. Second, and decisively, collapse still happens and the detector still misses it at every scale, trigram diversity falls in every run (e.g. RAID 0.86 $\rightarrow$ 0.27, HC3 0.86 $\rightarrow$ 0.24 at 15,000 documents) while the per-document detector score stays flat-to-falling (from **14**–**16** down to ~**9**) and estimated contamination remains 0% in all six runs. At 15,000 documents the vocabulary is *largely preserved* yet n-gram diversity collapses anyway: distributional *structure* degrades even when the word list does not, and the document detector is blind to both. The distinction is invariant to two orders of magnitude of scale and to the choice of seed corpus.

Does it hold for a modern architecture? distilGPT-2 dates from 2019; a reviewer may ask whether the finding survives on a contemporary LLM. We repeated the neural experiment with **Qwen2.5-0.5B** (a 2024 Llama-style model), LoRA-fine-tuned

Table 9: Recursive model collapse: trigram and neural (distilGPT-2), per generation.

Experiment	Gen	Vocab	Trigram div.	Type-token	Ref. ppl	Detector
Trigram	g0	6707	0.943	0.212	—	16.7
Trigram	g6	1287	0.270	0.113	—	11.9
Neural	g0	6707	0.943	—	69.2	16.7
Neural	g1	1708	0.926	—	25.7	16.3
Neural	g2	1387	0.840	—	9.8	19.8
Neural	g3	1196	0.768	—	9.5	21.7
Neural	g4	1054	0.695	—	8.2	24.8
Neural	g5	1012	0.639	—	7.4	27.1

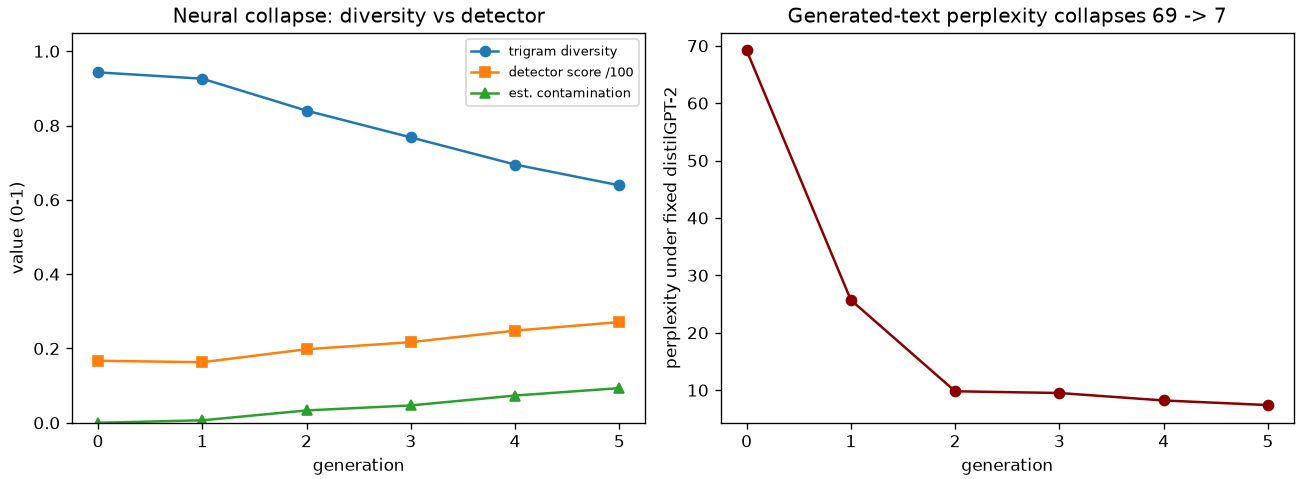


Figure 8: Neural collapse (distilGPT-2 retrained on its own output). Left: diversity and detector score per generation. Right: perplexity of generated text under a fixed reference model, 69 to 7.

recursively on its own output for three generations on the same commodity MPS hardware. The result is the same, and sharper: reference perplexity collapses monotonically 37 → 2.9 and vocabulary falls -87%, yet the per-document detector stays flat (17.6 → 18.4) with estimated contamination of just 2%. The distinction is not an artifact of an old, tiny model, it holds for a current architecture.

Unified conclusion. Across all three generative processes and all scales, distribution-level metrics catch collapse cleanly and monotonically while the per-document detector stays flat or sub-threshold (~91% of collapsed documents unflagged even in the distilGPT-2 case). We develop the methodological implication, that **per-document contamination detection is not a reliable collapse monitor**, in Section 6.3.

5.7 Comparison to Established Likelihood-Based Detectors

A fair question is how the deliberately cheap SyntheticTextProbe compares to established statistical detectors. We implement the two canonical GLTR statistics [Gehrmann2019] under GPT-2 (124M): the mean token log-probability, and the fraction of tokens that fall in the model's top-10 predictions (the "green" fraction). Both are zero-shot (no training). We do *not* run DetectGPT's perturbation-curvature test [Mitchell2023] or the commercial GPTZero [Tian2023]: the former needs many mask-fill perturbations and white-box access to the scoring model, the latter a paid API. Our GPT-2 log-probability detector belongs to the same likelihood family and serves as the runnable representative. Table 10 reports zero-shot AUC on HC3 (human vs. ChatGPT) and on pooled RAID (human vs. all eight generators).

Table 10. Zero-shot detection AUC (no training): GLTR statistics under GPT-2 vs. the SyntheticTextProbe heuristic. RAID is pooled across all eight generators.

A likelihood detector with white-box access to a reference model clearly outperforms our cheap heuristics in the zero-shot setting, 0.995 vs. 0.915 on HC3, and 0.862 vs. 0.724 on pooled RAID (the SyntheticTextProbe figures are consistent with the 0.912 held-out value in Section 5.1 and the 0.723 pooled value in Section 5.3). The SyntheticTextProbe is not a better detector; its advantages are transparency and zero dependencies: GLTR and DetectGPT require a scoring model's per-token probabilities, whereas the six heuris-

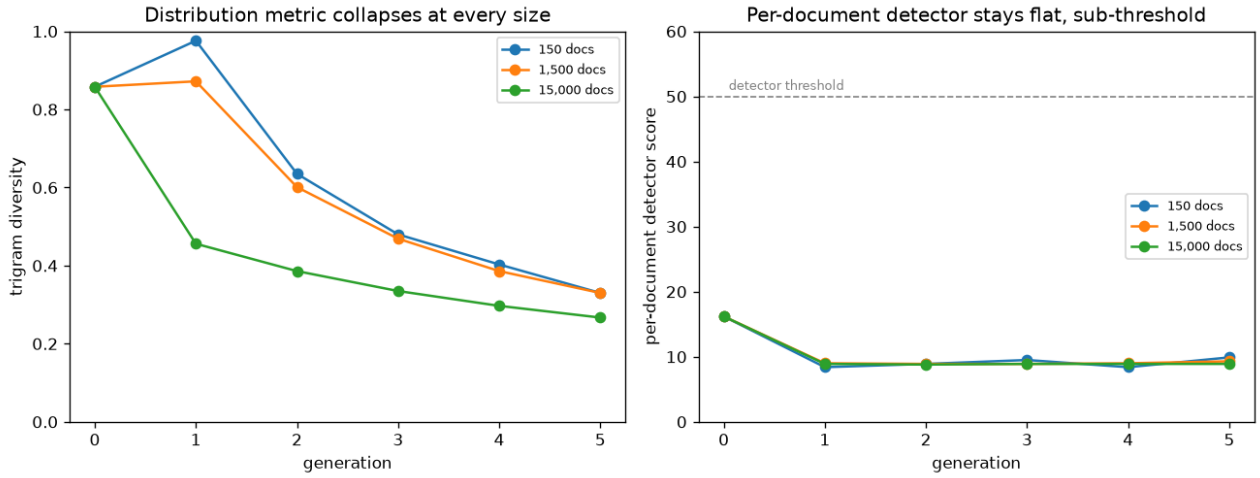


Figure 10: Collapse at scale (RAID seed; HC3 is similar). Left: trigram diversity falls at every per-generation corpus size (150 / 1,500 / 15,000 documents); larger corpora collapse more slowly but always collapse. Right: the per-document detector score stays flat and far below its threshold (dashed) at every size. The detection-vs-monitoring distinction is invariant to two orders of magnitude of scale.

Table 10: Zero-shot detection AUC: GLTR statistics under GPT-2 vs. the SyntheticTextProbe heuristic, on HC3 and pooled RAID.

Detector (zero-shot)	HC3 (vs ChatGPT)	RAID (vs 8 generators)
SyntheticTextProbe (six heuristics)	0.915	0.724
GLTR — GPT-2 log-probability	0.995	0.862
GLTR — GPT-2 top-10 rank	0.995	0.848

tics are plain text statistics. GPT-2's near-perfect HC3 score also reflects a family advantage, ChatGPT and several RAID generators are GPT-lineage models whose output GPT-2 finds highly probable.

Two points matter for this paper's thesis. First, the trained leave-one-model-out classifier (Section 5.4) recovers much of this gap on RAID using only an in-distribution *bigram* perplexity, no large model required, reaching 0.915 ± 0.006 held-out. Second, and more important: **every detector in Table 10 is a per-document detector**, so the central result of Section 5.6 applies to all of them equally. A stronger document-level detector does not become a collapse monitor. The very GPT-2 likelihood signal that wins here, when applied at the *distribution* level as a corpus's reference perplexity, is exactly the statistic that tracks collapse (69 → 7, Section 5.6). The detection-versus-monitoring distinction is therefore about *what one measures over*, a single document versus a distribution, not about how strong the detector is.

6. Discussion

6.1 What Generalizes, and What Does Not

The leave-one-model-out results answer the central empirical question, can cheap statistics plus a small perplexity signal detect synthetic text from *unseen* generators?, in the affirmative, with qualifications. The trained classifier reaches mean held-out AUC 0.915 ± 0.006, significantly above perplexity alone, and holds up on every generator from gpt2 (0.834) to mistral-chat (0.981). That a classifier trained on seven generators transfers to an eighth is the result we stand behind most: the statistical fingerprints of synthetic text are partly generator-agnostic. The lift over perplexity is modest and perplexity is the workhorse feature (Section 5.4), the cheap statistics are a useful supplement, not a replacement.

The *zero-shot* heuristic, by contrast, transfers unevenly (Section 5.3): strong on RLHF-tuned chat models, near chance on gpt2 base completions, which lack the canned-filler, low-burstiness profile the signals target. This is a limitation of fixed hand-set weights, not the feature set, the trained classifier recovers strong performance on gpt2 (0.834) where the zero-shot detector collapses to 0.566.

6.2 A Bigger Model Is Not Automatically a Better Feature

The most surprising negative result (Section 5.5) deserves emphasis: substituting a real, pretrained GPT-2 (124M) perplexity for the bigram language model *lowered* held-out detection on average (classifier 0.895 vs. 0.924 at the reference seed). The comparison is **not apples-to-apples**, the bigram is re-fit in-distribution per fold while GPT-2 is fixed and zero-shot, and two readings hold at once: the in-distribution bigram is the stronger feature *as measured here*, yet reaching roughly **0.86–0.90** with *zero corpus fitting* is arguably the more deployable signal where no in-distribution reference exists. When the perplexity feature weakens (GPT-2), the classifier leans *more* on the cheap statistics (lexical_diversity 1.24→1.46, repetition 0.58→0.90), it compensates exactly as a transparent feature-weighted system should. The lesson is narrow but real: a larger language model is not automatically a better detection feature than a tiny, domain-fitted one.

6.3 Per-Document Detection Is Not Distribution-Level Collapse Monitoring

The collapse experiments (Section 5.6) draw the sharpest conceptual line in the paper, summarized in Figure 9: AI-text detection and collapse monitoring are different problems, and different signals answer them. We did not discover model collapse, that phenomenon was established by [Shumailov2023]. Our contribution here is to instrument collapse and ask whether a per-document AI-text detector tracks it. It largely does not.

Under recursive retraining, distribution-level metrics degrade unambiguously while the per-document detector stays blind. For the trigram model, vocabulary falls −81% and trigram diversity 0.94→0.27, yet the detector score is flat-to-falling (16.7→11.9) and contamination barely moves (0→1%). The neural case (distilGPT-2) is kinder but still damning: reference perplexity collapses monotonically 69→7, yet the detector, though it does rise (16.7→27.1), never crosses its threshold, leaving ∼91% of collapsed documents unflagged (Section 5.6).

The unified conclusion is methodological. **Distribution-level metrics, vocabulary size, distinct-n-gram diversity, and reference perplexity, catch collapse cleanly and monotonically; per-document AI-text detection catches it weakly (neural) or not at all (trigram).** These are different problems. A per-document detector answers "is this one text synthetic?"; collapse monitoring answers "is this corpus's distribution narrowing over time?" A high-quality detector for the first question can be nearly blind to the second. Anyone hoping to use a deployed per-document detector as a tripwire for training-data contamination should treat the perplexity-under-a-fixed-reference and corpus-diversity statistics, not the detector's per-document verdict, as the operative signal.

6.4 Negative Results as Contributions

Three of our findings are contributions because they are negative. (i) Fixed-weight heuristics transfer unevenly and fail near chance on base-model completions, so they cannot be sold as general-purpose detectors. (ii) A larger pretrained language model (GPT-2) is not automatically a better perplexity feature than a tiny in-distribution bigram. (iii) A competent per-document detector is not a *reliable* collapse monitor.

---

7. Threats to Validity

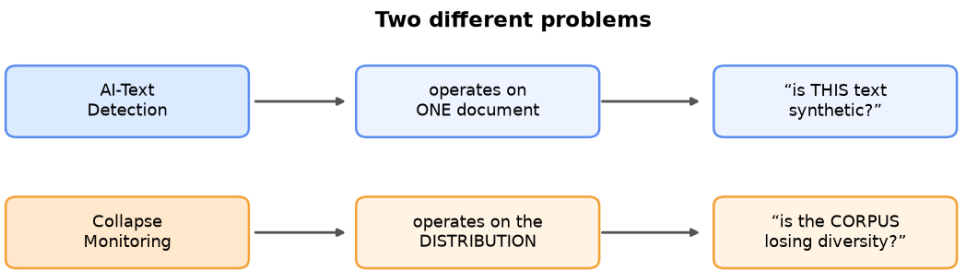
We state the scope of our claims and enumerate the conditions under which they could fail.

What we do *not* claim. We do not claim that the *exact* numerical behavior of any particular detector, its slope, the generation at which it crosses threshold, or the precise fraction of collapsed documents it flags, generalizes to every collapse process, dataset, or model scale. Our results demonstrate a *distinction*, not universal quantitative collapse rates.

What we *do* claim. The qualitative separation between document-level detection and distribution-level monitoring persists across everything we tested: three generative processes (a trigram LM, distilGPT-2, and the modern Qwen2.5-0.5B), two seed corpora from different domains (RAID and HC3 human text), and per-generation corpus sizes from 150 to 15,000 documents (Section 5.6). In every configuration, distribution-level metrics fall while the per-document detector stays flat or sub-threshold; larger corpora collapse more slowly, but the direction of every signal is invariant.

Specific threats.

- **Collapse processes.** Our collapse experiments use a trigram LM, distilGPT-2, and a modern Qwen2.5-0.5B (2024). Larger *frontier* models, diffusion or masked LMs, and full RLHF training loops may behave differently; recursively retraining models at that scale is beyond commodity hardware, so we make no claim about how fast they collapse.
- **Detector features are English-specific.** The six signals (filler phrases, tokenization, n-gram statistics) and the perplexity features are tuned to English text and would need re-derivation for other languages or modalities.
- **Datasets may not represent future models.** RAID and HC3 are particular, now-somewhat-



Which signal answers which question

Signal / metric	Flag one AI document	Detect corpus collapse
Perplexity (under a fixed LM)	Good	Good
Per-document detector score	Good	Poor
Vocabulary size	Poor	Excellent
Distinct n-gram diversity	Poor	Excellent

Figure 9: The paper's central claim in one figure. AI-text detection operates on a single document; collapse monitoring operates on the distribution. Per-document statistics flag individual AI documents but miss collapse, while distribution-level statistics (vocabulary, n-gram diversity) detect collapse but say little about a single document.

dated corpora (eight generators, English, ~250-word excerpts). Results may not transfer to newer model families, other domains, or much longer or shorter texts.

- **Quantitative magnitudes are model-specific.** The absolute numbers in Section 5.6 are specific to these small models and seed corpora; the transferable claim is the *direction* (distribution metrics fall, the detector stays sub-threshold), not the magnitudes.
- **Perplexity and diversity are proxies for collapse.** We instrument collapse with vocabulary size, distinct-n-gram diversity, and reference perplexity, all *proxies* for distributional degeneration, not collapse itself. Other facets, factual drift, bias amplification, long-range incoherence, are not measured and could behave differently; the claim concerns the metrics we track.
- **Results may vary under different sampling strategies.** Collapse rate depends on decoding: temperature, top-p, and repetition penalty all change how quickly a model degenerates on its own output. We use fixed settings throughout; other strategies could shift the magnitudes, though we expect the document-versus-distribution distinction to persist.
- **Statistical scope.** We provide multi-seed confidence intervals and a paired significance test for the headline LOMO result only (Section 5.4.1); the GPT-2-feature variant, the cross-model table, and the collapse runs are single-seed point estimates and should not be over-read at the level of small differences.
- **The perplexity comparison is not apples-to-apples.** The bigram LM is re-fit in-distribution per fold while GPT-2 perplexity is fixed and zero-shot, so the comparison in Sections 5.4–5.5 reflects information conditions as much as model size.
- **The detector is an un-tuned heuristic.** Its six weights and the threshold of 50 are fixed by hand, not tuned per dataset; the duplicate-similarity signal is inert (zero standardized coefficient, no reference corpus supplied) and a no-op in these experiments.

Construct validity. One might object that a *better* detector would track collapse. We think not: Section 5.7 shows that a strong, white-box GLTR-style likelihood detector beats our heuristics at the document level yet remains a per-document method, and the same likelihood statistic becomes a collapse signal only when computed over the *distribution* (a corpus's reference perplexity). The limitation is about the *unit of measurement*, not the quality of the detector, which is why we expect the distinction to hold for detectors stronger than any we tested.

8. Conclusion and Future Work

8.1 Conclusion

We studied whether cheap, interpretable text statistics plus a small perplexity signal can detect synthetic text across generators, and how per-document detection relates to distribution-level model collapse. Three findings anchor the paper. First, a transparent classifier over six heuristic signals and one perplexity feature **generalizes to unseen generators** (mean leave-one-model-out AUC 0.915 ± 0.006, significantly above perplexity alone), with perplexity as the workhorse feature and the cheap statistics a modest but real supplement. Second, a **larger pretrained model is not automatically a better feature**: a frozen GPT-2 perplexity underperformed a tiny in-distribution bigram, though the frozen signal is the more deployable one. Third, the result we most want remembered, **per-document AI-text detection is not a reliable collapse monitor**: under recursive retraining, distribution-level metrics collapse cleanly and monotonically while the per-document detector stays flat or sub-threshold (~91% of collapsed documents unflagged), and the same limitation applies to a stronger GLTR-style detector. Detection asks whether *this* text is synthetic; monitoring asks whether the *distribution* is narrowing, and they call for different signals. The entire core runs on the Python standard library; only the GPT-2 experiments need a heavier, isolated environment.

8.2 Future Work

Several directions follow directly from the limitations above.

- **Variance and significance.** Extend the multi-seed analysis already done for the headline LOMO result (Section 5.4.1) to the GPT-2-feature variant, the cross-model table, and the collapse experiments, so that every reported effect carries a confidence interval.
- **A fair perplexity comparison.** Compare the bigram language model and a neural language model under matched conditions, either both fixed/zero-shot or both fit in-distribution, to separate the effect of model capacity from the effect of in-distribution fitting.
- **Newer and larger generators, more domains and languages.** Extend the evaluation beyond the eight RAID generators and English to more recent model families, longer documents, and additional domains and languages, testing how far the generalization observed here holds.
- **Activating and replacing the inert signal.** Supply a reference corpus of known model outputs so that duplicate_similarity becomes informative, and explore additional cheap signals to replace or complement the no-op feature.
- **From detector to monitor.** The clearest next step is a dedicated *distribution-level* collapse monitor. Beyond the surface statistics used here, future work should investigate **embedding-space diversity** and **semantic entropy** as collapse signals, validated on **large-scale web corpora** rather than small seed sets, and evaluated as a contamination tripwire, rather than aggregating per-document detector verdicts.
- **Toward larger collapse studies.** Where compute allows, instrument recursive retraining of larger models to test whether the trigram-flat / neural-sub-threshold pattern persists as model capacity grows.

References

[Dugan2024] L. Dugan et al. "RAID: A Shared Benchmark for Robust Evaluation of Machine-Generated Text Detectors." *Proceedings of ACL*, 2024. arXiv:2405.07940.

[Gehrmann2019] S. Gehrmann, H. Strobelt, A. M. Rush. "GLTR: Statistical Detection and Visualization of Generated Text." *Proceedings of ACL (System Demonstrations)*, 2019.

[Guo2023] B. Guo et al. "How Close is ChatGPT to Human Experts? Comparison Corpus, Evaluation, and Detection." arXiv:2301.07597, 2023. (HC3 dataset.)

[Mitchell2023] E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, C. Finn. "DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature." *Proceedings of ICML*, 2023. arXiv:2301.11305.

[Radford2019] A. Radford et al. "Language Models are Unsupervised Multitask Learners." OpenAI Technical Report, 2019. (GPT-2.)

[Sadasivan2023] V. S. Sadasivan, A. Kumar, S. Balasubramanian, W. Wang, S. Feizi. "Can AI-Generated Text be Reliably Detected?" arXiv:2303.11156, 2023.

[Sanh2019] V. Sanh, L. Debut, J. Chaumond, T. Wolf. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter." arXiv:1910.01108, 2019. (distilGPT-2 lineage.)

[Shumailov2023] I. Shumailov, Z. Shumaylov, Y. Zhao, Y. Gal, N. Papernot, R. Anderson. "The Curse of Recursion: Training on Generated Data Makes Models Forget." arXiv:2305.17493, 2023. (Journal version: "AI models collapse when trained on recursively generated data," *Nature* 631, 2024.)

[Tian2023] E. Tian et al. “GPTZero.” Commercial AI-text detector, 2023. (Cited as a deployed per-document detector.)

Appendix A: Controlled Sanity Harness

The repository includes a separate **controlled** harness in which synthetic stand-in texts are constructed to differ from “human” texts on exactly the six signals the detector measures. Because the data are manufactured to be separable on those axes, it yields near-ceiling numbers (0.93–1.00 accuracy, ~1.0 self-contamination catch rate, 0% false positives). These outcomes are **tautological**, they verify the implementation behaves as constructed and say nothing about real text. All performance claims in this paper rest on the real HC3 and RAID results of Section 5, not on this harness; the harness figures live in the code release.