large scale and show that it significantly reduces the detectability of outputs. Following Keskar et al. (2019), we use $\theta = 1.2$ for our experiments.[3]

3.6 Adversarial Attacks

When selecting adversarial attacks, we assume that our adversary has exactly one query and no knowledge of the detector. Thus, we include the following 11 black-box, query-free attacks as opposed to gradient-based methods:

1. **Alternative Spelling**: Use British spelling
2. **Article Deletion**: Delete ('the', 'a', 'an')
3. **Add Paragraph**: Put `\n\n` between sentences
4. **Upper-Lower**: Swap the case of words
5. **Zero-Width Space**: Insert the zero-width space `U+200B` every other character
6. **Whitespace**: Add spaces between characters
7. **Homoglyph**: Swap characters for alternatives that look similar, e.g. `e` $\rightarrow$ `е` (`U+0435`)
8. **Number**: Randomly shuffle digits of numbers
9. **Misspelling**: Insert common misspellings
10. **Paraphrase**: Paraphrase with the fine-tuned T5-11B model from Krishna et al. (2023)
11. **Synonym**: Swap tokens with highly similar BERT (Devlin et al., 2019) candidate tokens

Following recommendations from Dyrmishi et al. (2023), we manually reviewed our data to ensure that adversarial attacks were inconspicuous. Thus, for each attack only a small percentage of the total available mutations were applied. For details on mutation percentages for each attack as well as other implementation details, see Appendix E.4.

3.7 Post-Processing

After all generations were completed, we removed prompts and left only the generated output. We then filtered out failed generations and balanced the dataset such that each human-written document has exactly one corresponding generation per model, decoding strategy, and adversarial attack.

4 Dataset

4.1 Statistics

The non-adversarial portion of the RAID dataset consists of 509,014 generations and 14,971 human-written documents for a total of 6,287,820 texts when including adversarial attacks. On average, models are more repetitive than humans as measured by Self-BLEU (Zhu et al., 2018) and typically generate shorter passages (see Table 3). The mean perplexity is lower for models than humans, according to LLaMA 7B and GPT 2 XL.

4.2 Release Structure

To accompany the RAID dataset we also release an official public leaderboard[4] which will host the results from our analysis alongside other detector results submitted by public contributors. The leaderboard is split up into two sections—one for

[3] Only open-source models provide access to a repetition penalty. OpenAI and Cohere instead offer slightly different "frequency" and "presence" penalties (see Appendix E.5). Thus we vary repetition penalty only for open-source models.

[4] https://raid-bench.xyz/leaderboard

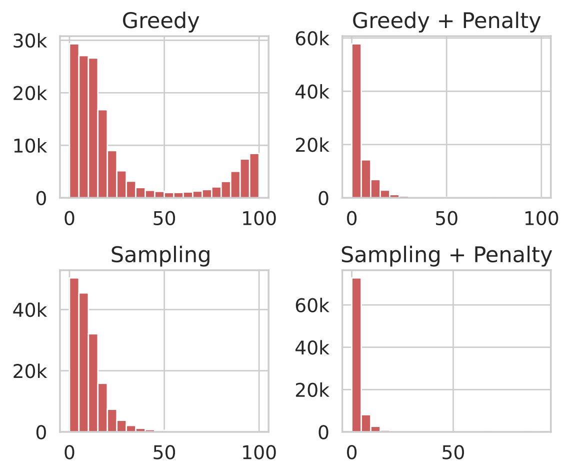

Figure 3: Histogram of examples (y-axis) grouped by their repetitiveness measured via SelfBLEU score (x-axis). We see that both random sampling and repetition penalty greatly reduce repetitiveness for all models.

Model	Num. Gens	Toks	Self-BLEU	PPL -L7B	PPL -G2X
Human	14971	378.5	7.64	9.09	21.2
GPT 2	59884	384.7	23.9	8.33	8.10
GPT 3	29942	185.6	13.6	3.90	8.12
ChatGPT	29942	329.4	10.3	3.39	9.31
GPT 4	29942	350.8	9.42	5.01	13.4
Cohere	29942	301.9	11.0	5.67	23.7
(+ Chat)	29942	239.0	11.0	4.93	11.6
Mistral	59884	370.2	19.1	7.74	17.9
(+ Chat)	59884	287.7	9.16	4.31	10.3
MPT	59884	379.2	22.1	14.0	66.9
(+ Chat)	59884	219.2	5.39	7.06	56.3
LLaMA	59884	404.4	10.6	3.33	9.76
Total	509k	323.4	13.7	6.61	23.8

Table 3: Statistics for the generations in the base dataset without adversarial attacks. **PPL-L7B** refers to mean perplexity according to LLaMA 7B and **PPL-G2X** refers to mean perplexity according to GPT 2 XL.

those who self-report having trained on the RAID dataset and one for those who do not. This is important to ensure that a clear distinction is made between detectors that are generalizing to out-of-domain data and those that are not.

To ensure fair competition, 10% of the RAID dataset is released without labels for use as the official hidden test set. We provide scripts to easily run detectors on this test set and calculate accuracy with respect to the hidden labels. Users can then submit their outputs to the leaderboard via a pull request (see Appendix D). We hope that this infrastructure encourages more comparison and shared evaluation of detectors.

4.3 RAID-extra

In addition to the over 6M+ generations in the core RAID train and test sets, we also release "RAID-extra", an additional dataset consisting of 2.3M generations from three extra domains not included in the main benchmark: Python Code, Czech News, and German News. In Appendix A we report evaluation results from our 12 detectors on RAID-extra and show that metric-based detectors perform surprisingly well on these types of unusual domains.

RAID-extra is the largest and most challenging dataset of generated code and multilingual text ever released. We hope it will be of value to the academic community.

5 Detectors

5.1 Detector Selection

We evaluate detectors from three categories: neural, metric-based, and commercial. Neural detectors typically involve fine-tuning a pre-trained language model such as RoBERTa (Liu et al., 2019) while metric-based detectors typically compute some metric using the output probabilities of an existing generative model. In contrast, commercial detectors tend to provide some documentation of their performance but disallow direct access to the models. We tested the following:

(i). **Neural**: RoBERTa-Base (GPT2), RoBERTa-Large (GPT2), RoBERTa-Base (ChatGPT), RADAR

(ii). **Metric-Based**: GLTR, Binoculars, Fast DetectGPT, LLMDet

(iii). **Commercial**: GPTZero, Originality, Winston, ZeroGPT

	τ=0.25	τ=0.5	τ=0.75	τ=0.95
R-B GPT2	8.71%	6.59%	5.18%	3.38%
R-L GPT2	6.14%	2.91%	1.46%	0.25%
R-B CGPT	21.6%	15.8%	15.1%	10.4%
RADAR	7.48%	3.48%	2.17%	1.23%
GLTR	100%	99.3%	21.0%	0.05%
F-DetectGPT	47.3%	23.2%	13.1%	1.70%
LLMDet	97.9%	96.0%	92.0%	75.3%
Binoculars	0.07%	0.00%	0.00%	0.00%
GPTZero	0.03%	0.00%	0.00%	0.00%
Originality	0.47%	0.25%	0.17%	0.07%
Winston	0.75%	0.55%	0.38%	0.21%
ZeroGPT	1.71%	1.42%	1.21%	0.90%

Table 4: False Positive Rates for detectors on RAID at naive choices of threshold (τ). We see that, for open-source detectors, thresholding naively results in unacceptably high false positive rates.

Unlike Li et al. (2024), we do not train our own neural models on our dataset because we wish to investigate the generalization ability of off-the-shelf detectors. For the metric-based detectors, we chose to use the default generative model in each repository to emulate the most realistic use-case.[5]

5.2 Detector Evaluation

Detectors work by taking in a sequence of tokens and outputting a scalar score. In order to convert this score to a binary prediction, we must select a scalar threshold τ such that if the score $s \geq \tau$ the sequence is predicted to be machine-generated.

In our work, we select a threshold for each model such that the resulting false positive rate of the detector is 5%. In practical terms, accuracy at a fixed FPR of 5% represents how well each detector identifies machine-generated text while only misclassifying 5% of human-written text. Our work is one of the first shared resources to fix and disclose FPR, following the rise of this evaluation paradigm in recent robustness research (Hans et al., 2024; Krishna et al., 2023; Soto et al., 2024).[6]

6 Findings

Finding 1: Default False Positive Rates (FPRs) of open-source detectors are dangerously high When applying a detector to a piece of text, it is important to decide on a threshold (τ) to use for binary classification. The principled way to determine this is to use a set of in-domain data to

[5]See Appendix F for more details on each detector tested

[6]See Appendix B for a discussion of why we chose this paradigm instead of the traditional precision/recall/F1 score

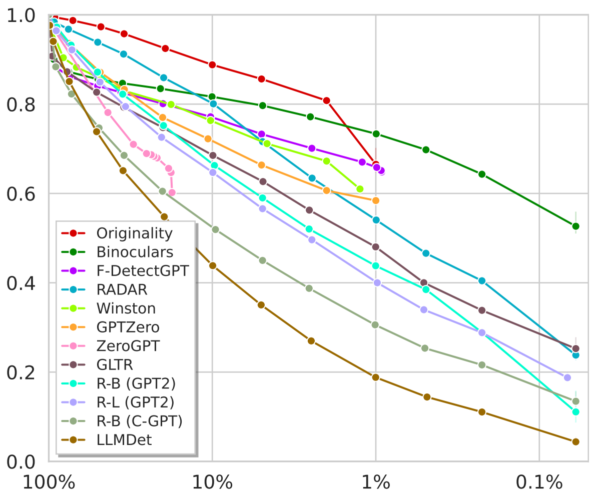

Figure 4: Detection accuracy (y-axis) vs. False Positive Rate (x-axis) for all detectors. We see that Binoculars works significantly better than other detectors at low FPR and that few detectors can operate at FPR<1%.

calibrate the threshold to a specific point along the receiver-operating curve. However, this is cumbersome and requires a set of readily accessible human-written data. Thus, in practice it is common to simply use some seemingly sensible default value (such as 0.5) for the threshold without further investigation.

In Table 4 we report the false positive rates of our detectors for various commonly chosen thresholds. We see that open-source detectors, especially metric-based detectors, exhibit dangerously high false positive rates when using these naive thresholds. On the contrary, closed-source detectors seem to be calibrated fairly well, with none having an FPR above 1.7%. Given this result, we advise practitioners to take care not to use naive-yet-sensible values for their detectors and instead calibrate detectors on in-domain data before using them.

Following this advice, for the remainder of the Findings section, we will be exclusively using thresholds that were calibrated to a FPR of 5% on the human-written portion of the RAID dataset (see Appendix C).

Finding 2: Detector accuracy varies substantially depending on target False Positive Rate In Figure 4 we report the results of an experiment where we varied the classification threshold and plotted detector accuracy vs. false positive rate. We found that our detectors were capable of achieving the high accuracies cited in many viral reports, but only at similarly high FPR. Some detectors failed to achieve the lowest FPR we tested, plateauing at 16.9% (ZeroGPT), 0.88% (FastDetectGPT), and 0.62% (Originality). Most detectors dropped steeply as FPR decreased from 100%, but Binoculars (Hans et al., 2024) was particularly strong at low FPR. Since detector accuracy varies so much with FPR, explicitly calibrating and reporting FPR is crucial for comparable, informative, and reproducible detection studies.

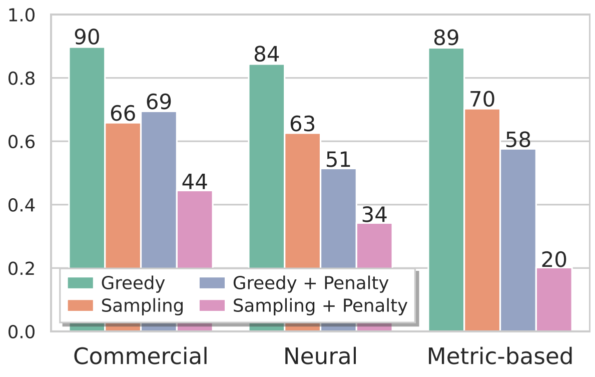

Figure 5: Accuracy at FPR=5% (y-axis) vs. decoding strategy across our three detector classes on non-adversarial text. We see that repetition penalty greatly reduces accuracy in both greedy decoding and sampling.

Finding 3: Repetition penalty drastically hurts accuracy for all detectors As shown in Figure 5, we observe a consistent pattern across detectors, detector categories, generators, and domains: adding a repetition penalty decreases accuracy by up to 32 points regardless of decoding strategy. The decoding strategy matters as well. Detectors in each category, across domains and generators, perform substantially better on greedy decoding than random sampling, even when taking repetition penalty into account.

This pattern is especially concerning because past studies have largely overlooked variations in decoding strategy when evaluating detectors. Furthermore, none have reported results on repetition penalty before our study. Since sampling with repetition penalty results in text that often sounds more human-like (Keskar et al., 2019), exposing these patterns is critical for reliable detection.

There are many possible penalties and decoding strategies one could use when generating text such as contrastive decoding (Li et al., 2023), eta and epsilon decoding (Hewitt et al., 2022), and typical sampling (Meister et al., 2023)—all of which are likely to reduce detector accuracy. We highly encourage robustness studies and open evaluations to investigate how well detectors can generalize to