Technical Report on the Checkfor.ai AI-Generated Text Classifier

Bradley Emi[*]
Checkfor.ai
Brooklyn, New York
bradley@checkfor.ai

Max Spero*
Checkfor.ai
Brooklyn, New York
max@checkfor.ai

February 27, 2024

Abstract

We present the Checkfor.ai text classifier, a transformer-based neural network trained to distinguish text written by large language models from text written by humans. Checkfor.ai outperforms zero-shot methods such as DetectGPT as well as leading commercial AI detection tools with over 9 times lower error rates on a comprehensive benchmark comprised of ten text domains (student writing, creative writing, scientific writing, books, encyclopedias, news, email, scientific papers, short-form Q&A) and 8 open- and closed-source large language models. We propose a training algorithm, hard negative mining with synthetic mirrors, that enables our classifier to achieve orders of magnitude lower false positive rates on high-data domains such as reviews. Finally, we show that Checkfor.ai is not biased against nonnative English speakers and generalizes to domains and models unseen during training.

Keywords AI Text Detection, Large Language Models

1 Introduction

AI-generated text detection has become an increasingly important problem to solve. Large language models (LLMs) such as GPT-3, (Brown et al., 2020) GPT-3.5, and ChatGPT series (OpenAI, 2023a), Gemini (Google, 2023), and several open source models are now capable of producing fluent text that even trained experts are not able to distinguish perfectly (Casal and Kessler, 2023). As a result, several research and commercial solutions have recently been developed, notably TurnItIn and GPTZero (Tian and Cui, 2023), among numerous others.

However, the shortcomings of existing AI detection methods have been well-documented (Weber-Wulff et al., 2023). Commercial AI detectors such as TurnItIn have demonstrated inadequate accuracy for the context of academic plagarism - primarily due to high false positive rates (Vanderbilt, 2023). Additionally, zero-shot AI detection methods have been shown to be biased against nonnative English writing (Liang et al., 2023). For many applications, such as academic integrity enforcement, reducing false positive rates is a key barrier to the adoption of AI detection tools.

Our technical approach is motivated by the need for robust detection methods with extremely low false positive rates that can be used in production settings. Perplexity based methods - such as DetectGPT, Sniffer and DetectLLM - fail on human-written documents that happen to be in the training set of LLMs, such as the Declaration of Independence. On the other hand, deep learning based classifiers are not robust to out-of-domain examples: as shown by DetectGPT and many others, these detectors perform poorly when run on text that differs significantly from their training distributions.

In this technical report, we first present a high level overview of our results as compared to two commonly used commercial detectors, GPTZero and Originality AI, as well as one of the most popular academic methods, DetectGPT (Mitchell et al., 2023). In Section 3, we explain our methodology, as well as the motivation for our technical approach: the saturation of scaling laws. We believe that the only way to train a production-level AI classifier is by training it on a large-scale dataset comparable in size to the datasets that are used to train modern LLMs. However, we show that a

---

[*]Equal contribution. Correspondence to info@checkfor.ai

naive approach, simply training the classifier on a multi-million example dataset out of the box, will not work. Not only is this method extremely cost ineffective, so much so that to actually run the experiment would be impossible, but we demonstrate that at a certain critical threshold, adding additional randomly sampled examples does not improve accuracy. From our scaling law experiments, we can infer that the naive baseline would cap out in performance as well.

The solution to the scaling law saturation issues is to resolve poorly conditioned optimization (e.g. gradient collapse) via a learning curriculum based on active learning and hard negative mining. We detail a novel algorithm for training AI-text generation models that scales to web data by efficiently selecting and generating novel AI examples based on offline inference results from a partially trained classifier. We show that this approach yields superior results and enables generalization patterns that were previously not thought to be possible with a deep learning based AI-generated text classifier.
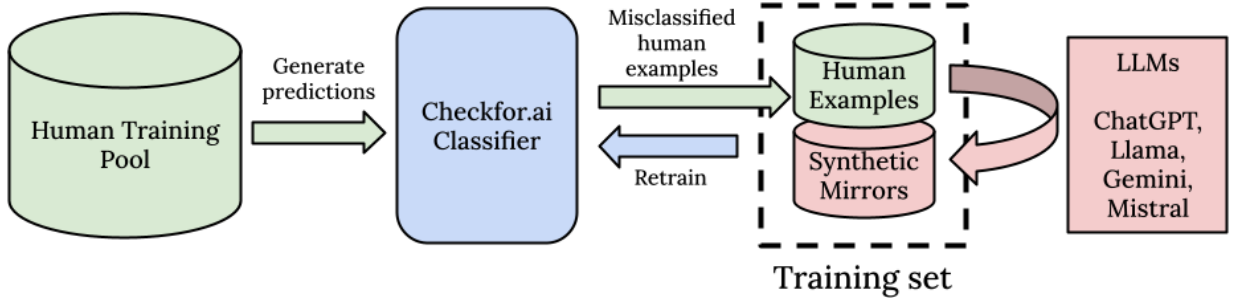
2   Algorithm

Figure 1: Training process for the Checkfor.ai AI-generated text classifier. An initial classifier predicts on a large training pool of human examples, identifying false positives which are then added to the training set and mirrored by LLMs.

2.1   Training Algorithm

Our model is a slightly modified transformer-style architecture (Vaswani et al., 2017). The classifier is trained on a mixture of human examples and synthetic examples generated by LLMs to closely match the content of the human examples, using a method called mirror prompting that we detail in Section 4.2.

We consider our dataset and training method to be the primary reason for our improvement in performance over other state-of-the-art methods.

It would be exorbitantly cost-prohibitive to generate tens of millions of synthetic mirrors to match the human examples in our training pool. Additionally, we find that training a neural network with a standard loss function reaches convergence before the first epoch concludes. We hypothesize that the reason for this early convergence is that the vast majority of examples are trivially easy for the network to classify: given only a few examples, the network is able to distinguish most AI examples from most human examples. As a result, the network converges to a "spiky" loss function– where most batches of input have close to zero loss and therefore zero gradient, drastically reducing the efficiency per unit compute.

To solve this early convergence issue, we treat the problem as a coreset selection problem, where the optimization objective is to search for difficult examples that resuscitate the gradient signal to the network. After initial training, we use hard negative mining to ensure that we add only increasingly difficult examples to the training set to classify. Our approach also shares commonalities with curriculum learning, which rather than uniformly sampling examples from the training set, suggests training the network with a time-dependent distribution that provides better conditioning for the loss function and therefore more well-behaved optimization. A schematic of our method is presented in Figure 1. We describe the dataset and training method in more detail in Section 4.
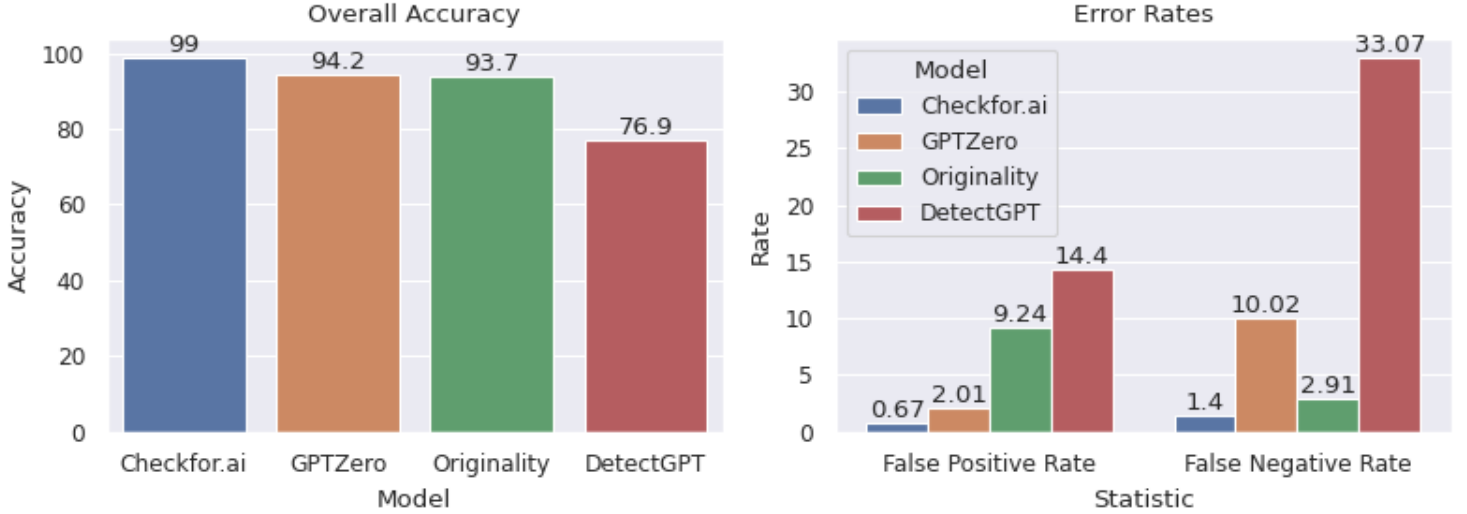
Figure 2: Overall results. (Left): Accuracy by detection method. (Right): False positive and false negative rates by detection method. Checkfor.ai has significantly higher accuracy than the next best methods, demonstrating state-of-the-art performance.

3 Results

3.1 Overview

We present a comprehensive benchmark on 1,976 documents against commercial AI detectors GPTZero and Originality.ai, as well as the zero-shot academic method DetectGPT. The size of the dataset was chosen to balance coverage of a wide variety of domains and appropriate sample size for evaluation, with cost-effectiveness and reproducibility. The benchmark size is such that, at the time of publication, it approaches the quantity of text one can scan with a monthly subscription to GPTZero or Originality. This benchmark is comprised of documents from eight LLMs - five commercial and three open-source - as well as ten different text domains.

Models	Domains
	Blog Posts
GPT 3.5 Turbo 0301	Scientific Writing
GPT 3.5 Turbo 1106	News
GPT 4 0613	Books
GPT 4 Turbo 1106	Student Writing
Google Gemini Pro	Email
Mistral 7B Instruct	Creative Writing
Mixtral 8x7B Instruct	Reviews
LLaMA 2 70B Chat	Q&A
	Wikipedia

Table 1: Models and text domains included in the benchmark. Each model is used to generate synthetic examples evenly across all domains.

We chose several versions of ChatGPT because it is the dominant LLM in terms of popularity and market share (GlobalData, 2023). While it is difficult to estimate the exact market share, based on social media analytics, ChatGPT represents 89% of the LLM market, while Bard is a distant second at 5%, with no other LLM surpassing 1%. Bard has been retired, but it has been replaced with Gemini– thus, we include Gemini Pro in our benchmark as well. We also include popular open-source models LLaMA 2 (Touvron et al., 2023) and two models from the Mistral team (Jiang et al., 2023) (Jiang et al., 2024) due to their popularity in the LLM open-source community.
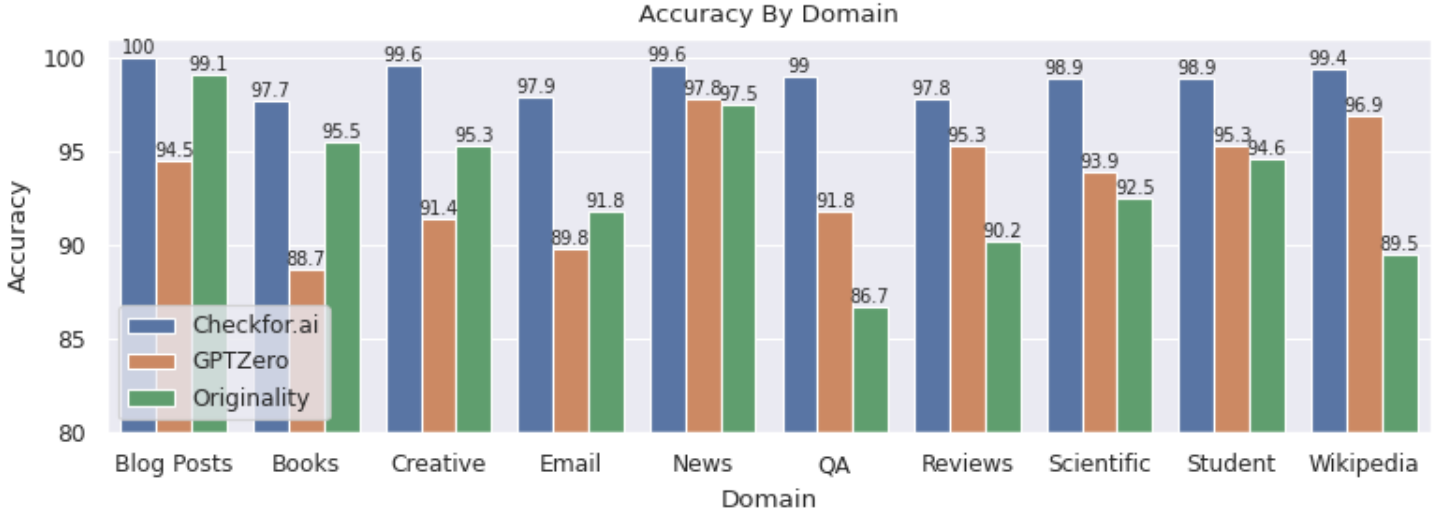
Figure 3: Accuracy by Domain. Checkfor.ai outperforms GPTZero and Originality on all 10 domains tested, demonstrating robustness to a wide variety of writing styles and formats.

Due to its significance in AI research, we also intend to include Claude by Anthropic (Anthropic, 2023) in a future version of this benchmark. However, we omitted it from this benchmark due to several observed issues getting Claude to respond correctly to our prompts and a high number of rejected requests. We felt that this would not be an accurate representation of model performance on Claude outputs due to the high amount of manual postprocessing that needs to be done before Claude outputs are clean enough for most practical applications.

For DetectGPT, we evaluate it in the most favorable black box setting for ChatGPT, using the best performing model from the large-scale study of cross detection (Mireshghallah et al., 2023), with 100 perturbations from T5-3B and using OPT-125M as the cross-detector.

3.2 Overall Performance

Checkfor.ai's text classifier is the only model that achieves production-ready levels of accuracy, false positive rate, and false negative rate. Our model is the most accurate at 99%, compared to commercial competitors which do not even clear 95%. Our false positive rate is better than the second best model, GPTZero, by a factor of 3, which achieving 7 times better negative error rate.

Notably, GPTZero's false negative rate is 10.02%– it is extremely biased towards predicting false negatives rather than false positives. This is important for educators, but seriously hampers the reliability of detection, compromising its ability to confidently authenticate that human text is actually human-written. Originality has the opposite issue– its false positive rate is 9.24%, which is simply too high to be practical. Human text comprises a vast majority of documents in most applications, with AI text still being a relatively novel occurrence, so with such a high false positive rate, most of the AI predicted documents will actually be human.

As DetectGPT performance is extremely poor compared to the commercial detectors on our benchmark, we omit results from DetectGPT for the remainder of the methods section.

3.3 Performance by Domain

We compare the performance of Checkfor.ai against GPTZero and Originality on 10 domains of text: blog posts, books, creative writing, email, news, Q&A, reviews, scientific writing, student writing, and Wikipedia. Creative writing is taken mostly from the Reddit WritingPrompts subreddit (Fan et al., 2018) as well as the Ghostbusters subset (Verma et al., 2023). Books are taken from Project Gutenberg. Email is taken from the Enron email dataset (Enron Corp and Cohen, 2015). QA is taken from a mix of sources, including Reddit ELI5, AskHistorians, AskScience (Fan et al., 2019), as well as the open source MedQUAD (Asma and Demner-Fushman, 2019) and FIQA datasets (fiq, 2018). Student writing is taken from a mix of sources, including the PERSUADE corpus (University et al., 2023), the Liang benchmark
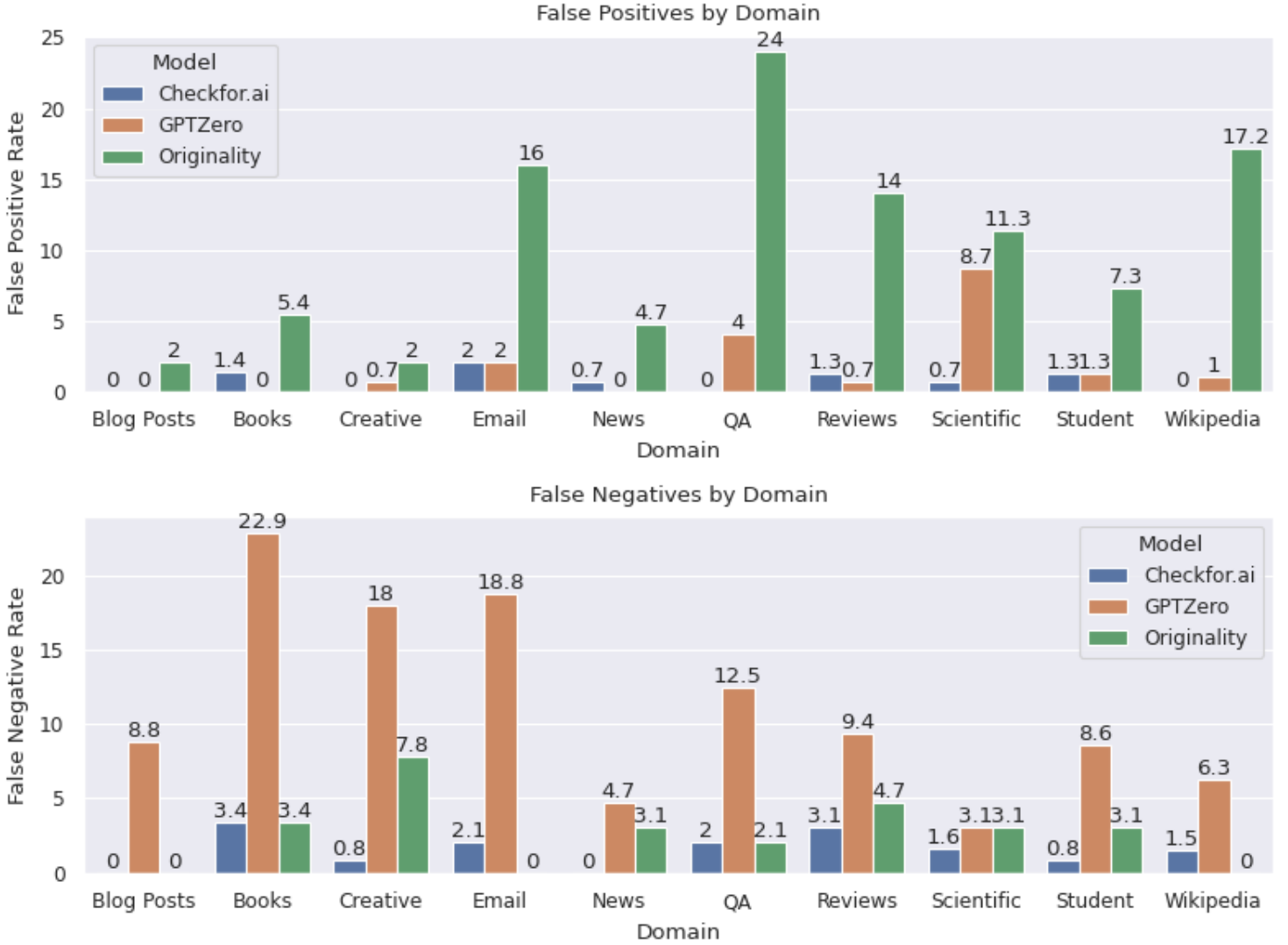
False Positives by Domain

False Negatives by Domain

Figure 4: False Positive and False Negative Rates by Domain. Other models show bias towards over- or under-predicting AI labels. Checkfor.ai is the only model that achieves both low FPR and FNR.

(Liang et al., 2023), and college admissions essays collected from various sources around the web. Reviews are taken from Amazon, Google Maps, and Yelp directly. In all cases, human data is verified to have been written prior to 2021 to avoid contamination from AI writing.

3.4 Performance by LLM

We also evaluate the performance of each classifier by its ability to detect AI documents from the different LLMs in the benchmark survey. A detector with a high false positive rate could game the accuracy metric simply by lowering the threshold, capturing lots of AI text at the expense of also classifying a lot of human text as AI as well. To normalize for the threshold and avoid each classifier being able to trade off false positives for false negatives, we first set a threshold such that each model has a fixed false positive rate of 1 percent. Then, given this threshold, we evaluate the recall: the fraction of AI documents able to be classified at that threshold. Using this metric, each classifier can be evaluated independently of the chosen threshold.

GPTZero does not use a threshold, rather it uses a ternary classification scheme between "HUMAN", "AI" and "MIXED". Unfortunately it does not expose the relative probabilities of each category in the exported output, so we use the hard

prediction from GPTZero instead. Since GPTZero has a false positive rate of 2 percent, these recall numbers are an optimistic upper bound on GPTZero's actual performance.
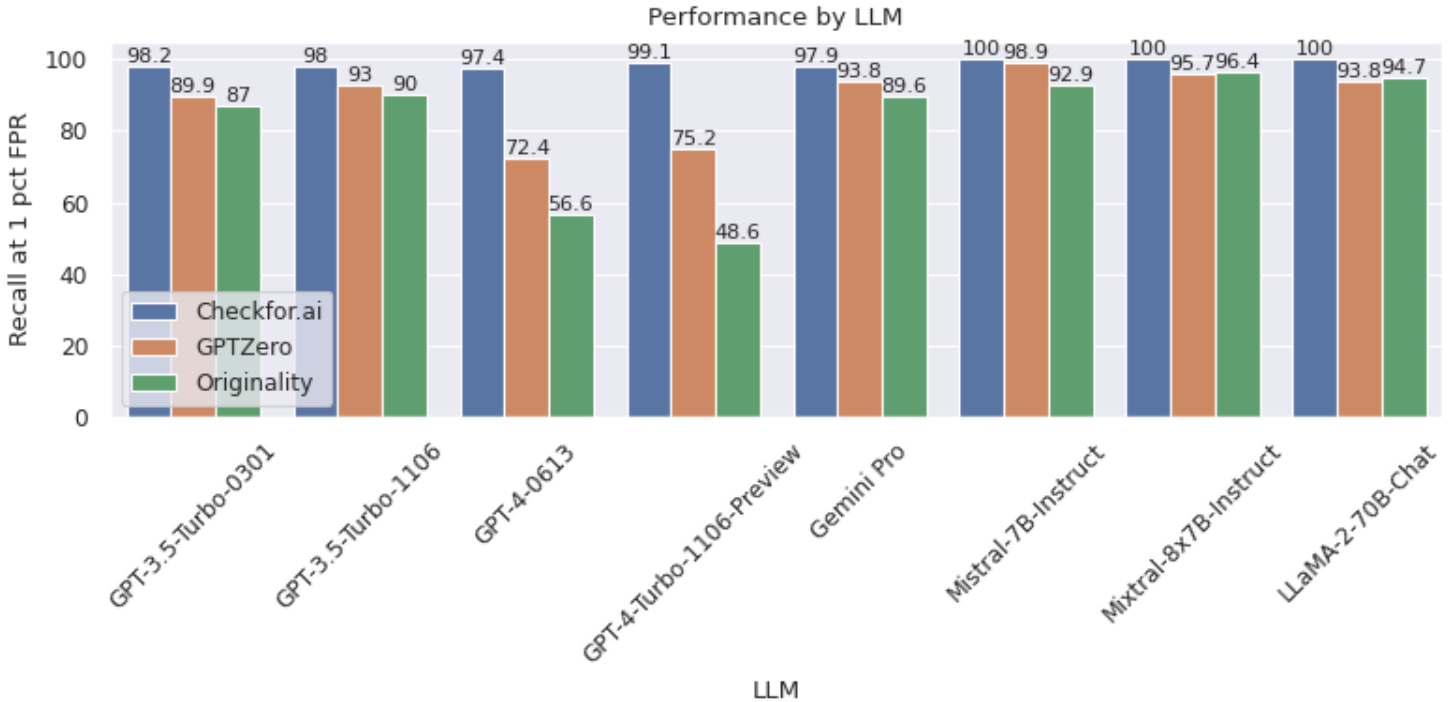

Figure 5: Recall at 1% FPR by LLM that generated the AI text. Checkfor.ai's performance remains strong on the most capable model, GPT-4, while the other models experience a severe degradation in performance.

We observe that Checkfor.ai is the only model that can achieve greater than 97% recall on all the LLMs tested. Interestingly, GPTZero performs best on the open-source models. This is likely due to the fact that perplexity and burstiness features can be used from the same model that generated the output– as exact computation of perplexity can only be done in the white-box setting with an open source model.

GPTZero and Originality completely fail when asked to detect AI content produced by GPT-4, both the latest full version of GPT-4 as well as GPT-4-Turbo. According to (Chakraborty et al., 2023), the GPT-4 family of models is the *most* difficult family to classify based on probability-based features, showing the limitations of the probability-based approaches to detect increasingly capable LLM output. This is the reason why Checkfor.ai uses standard deep learning techniques – so that the model can learn the underlying patterns of speech and voice created by advanced LLMs such as GPT-4, rather than relying on probability estimates from less capable models. Additionally, the recent release of GPT-4 may mean that it was not incorporated into the training sets of the other models.

We estimate that we are the only AI detection model that will continue to hold performance even as LLMs become more capable.

3.5 Performance on Nonnative English (ESL)

Perhaps due to their simplistic language, researchers (Liang et al., 2023) found that commercial LLM detectors are consistently biased against nonnative speakers (English as a Second Language, or ESL). To test this, the researchers used a benchmark of 91 essays from TOEFL (Test of English as a Foreign Language) to test several detectors.

We hold out the 91 TOEFL essays from our training set and evaluate Checkfor.ai on the benchmark. Due to the composition of our training set, we report a false positive rate of 0% on the TOEFL benchmark. We report results in Figure 6.

In October 2023, GPTZero published a response to this benchmark on their blog, titled "ESL Bias in AI Detection is an Outdated Narrative" (Edward Tian, 2023) along with an updated model. However, the new model still achieves a false positive rate of 7.7%, or 1.1% if "Possible AI Content" is generously labeled a negative.
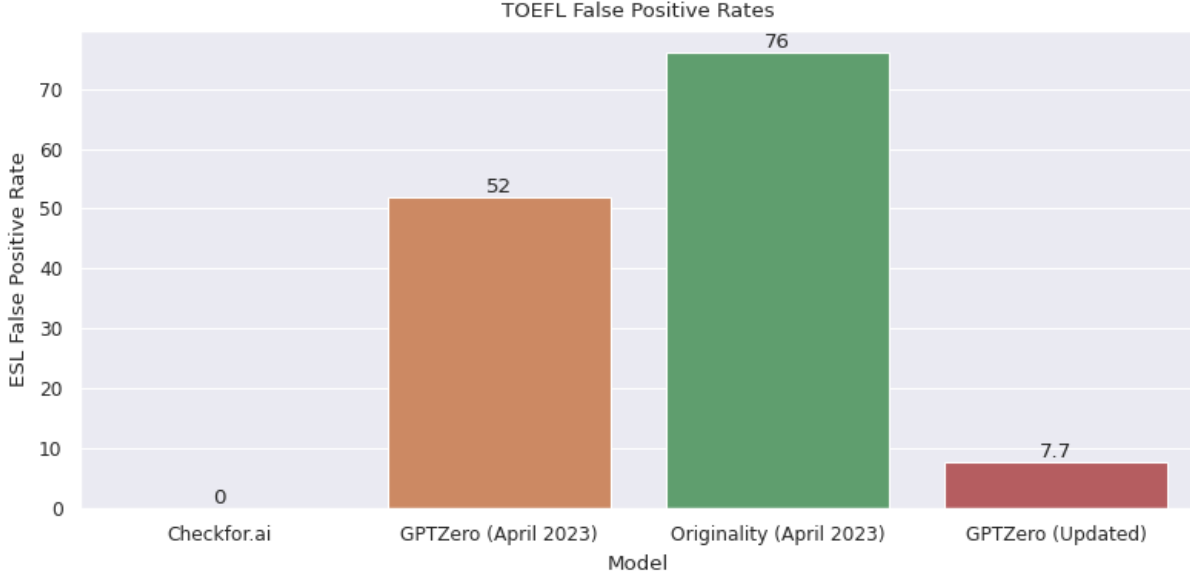
Figure 6: False positive rates on the TOEFL benchmark. Checkfor.ai is the only model to achieve zero false positive rate on this English as a Foreign Language benchmark.

3.6 Performance on Out-of-Distribution Examples

Deep learning based detectors are often criticized for being unable to generalize outside of the text distributions that they are trained on. DetectGPT (Mitchell et al., 2023) uses WMT and PubMedQA as example domains in which the RoBERTa based OpenAI text classifier fails to correctly distinguish text. While PubMedQA and WMT-like text are included in the domains (QA and general free-form text) of our classifier, one domain that is very difficult to source open data is email, due to the sensitive nature of the domain and the large amounts of personally identifiable information (PII) contained within it.

To show that our model is able to generalize outside of its training domain, we hold out all email from our training set and evaluate our model on the entire Enron email dataset, which was released publicly as a dataset for researchers following the extrication of the emails of all Enron executives in the legal proceedings in the wake of the company's collapse.

Our final model achieves a false positive rate of 0.8% on the Enron email dataset after hard negative mining, compared to our competitors (who may or may not have email in their training sets) which demonstrate a FPR of at least 2%. After generating AI examples based on the Enron emails, we find that our false negative rate is around 2%. We show an overall accuracy of 98% compared to GPTZero and Originality which perform at 89% and 91% respectively.

3.7 Performance on LLMs unseen during training

We evaluate our model on two new models released by OpenAI, `GPT-4-0125-Preview` as well as `GPT-3.5-Turbo-0125`. We note that the classifier has not seen a single example in the training set that comes from either of these two models. We create a benchmark dataset of 1830 and 1781 examples respectively and evaluate our model on the same metric as above, recall at 1% precision using the same threshold as in Section 2.4. For this experiment, we cannot fairly compare our model to our competitors in this setting, as we do not know whether or not those models have seen these during training.

We find that our model maintains strong performance on these two new models in Table 2, showing robustness to new OpenAI releases.

We are aware that we won't be able to train on the outputs of every model or even model family. However, our classifier must perform well on new and out-of-distribution open source models if we expect to serve as an effective safeguard for the internet. We evaluate the Checkfor.ai classifier on evaluation sets of 1300 examples from each of several models available in the Together AI API. The results in Table 3 show that our model effectively generalizes to open source models, with greater than 99.6% recall on all models tested.

LLM	Recall at 1% FPR
GPT-3.5-Turbo-0125	99.66%
GPT-4-0125-Preview	99.18%

Table 2: Checkfor.ai performance on OpenAI models unseen during training. Our model shows robustness to model updates without needing to regenerate new training data.

LLM	Recall at 1% FPR
OpenChat 3.5	100.00%
Qwen1.5-72B-Chat	99.93%
DeepSeek-Coder-33B-Instruct	99.75%
Yi-34B-Chat	99.68%
Vicuna-13B-v1.5	99.85%
MythoMax-L2-13b	99.61%
SOLAR-10.7B-Instruct-v1.0	99.61%

Table 3: Checkfor.ai performance on open source models unseen during training. Our model shows robustness to new models and model families trained on open source data.

3.8 Additional Benchmark Information

3.8.1 Dataset Sources

Our benchmark dataset comprises examples both from open-source text corpora as well as hand-picked examples from the internet. The importance of using hand-picked examples is that we do not know the training sets of other commercial detectors, but it is likely that they are trained on common open-source datasets available on HuggingFace and other common repositories. To accurately measure real-world performance and mitigate the risk of training set leakage, we collect text from many websites. For example, for the news domain, we use a fraction of the Reuters, BBC, and XSum datasets, but we also use a fraction directly copied and pasted from individual international, national, and local news websites. We publicly release our dataset [2] and can provide an exhaustive list of sources upon request.

3.8.2 AI-Generated Examples

We use standard prompts to create matching AI examples for each human example. For example, "Write a title for the following essay" followed by "Write an essay with the following title." We sometimes ask the LLM to use the first sentence of the document as the first sentence of the response, similar to (Mitchell et al., 2023).

We attempt to remove standard LLM boilerplate headers, such as "Sure! Here is an essay in response to your prompt:", by using the following heuristic. If the first paragraph starts with any of the following phrases: `Sure`, `Here is a`, `Title:`, `Abstract:`, `I have:`, or `I'm happy to help`, then we remove it. We reject any examples under 50 words long as these are typically implicitly rejected requests.

Finally, to avoid having any of the models using formatting rather than content to make their predictions, we remove all non-Unicode characters using `unidecode`, normalize all whitespace and quotations, and remove emojis. The result is a very clean benchmark of AI-generated examples that accurately reflect detection of AI-generated text based on content.

4 Method

4.1 Datasets

To train our model, we begin by considering a total pool of approximately 28 million confirmed human-written documents, described broadly in Table 4. The datasets are open source, freely available on the Internet, and importantly, are licensed for commercial use. We only consider datasets from 2021 and earlier to minimize the likelihood that any AI-generated text ends up mislabeled in our training set.

We exclude 4 million examples from our training pool as a holdout set to evaluate false positive rates following calibration on the above benchmark.

---

[2] `https://checkforai-public.s3.amazonaws.com/benchmark.csv`

Domain	Number of Examples
Creative Writing	300,000
Business and Product Reviews	15,000,000
Books	7,000,000
Scientific Papers	3,000,000
Wikipedia	1,000,000
News Articles	500,000
Q&A	1,000,000
Email	16,000
Student Writing	23,000
English as a Second Language (ESL)	165,000

Table 4: Human-Authored Examples Available by Domain

4.2 Mirror Prompts

A naive implementation of an AI text detection classifier could put N human documents into a training set and prompt an LLM for N AI-generated documents. A pitfall with this approach is that there are a number of features to overfit to - the topic chosen by the LLM, the length of the document, the default level of sophistication.

In order to prevent overfitting to any of these content-specific features, we design the AI side of the dataset to closely resemble the human side in style, tone, and semantic content. For each human example, we generate an AI-generated example that matches the original document on as many axes as possible, to ensure that our model learns to classify documents solely based on specific characteristics of LLM writing.

We define the term "mirror prompt" to be a prompt based on the original example that is used to generated a "synthetic mirror" or "mirror example." The goal of each mirror prompt is to generate an example that matches the topic and length of the original document.

If the original document is "<original review>", then a mirror prompt may look like this:

```
[Prompt] Write a <original review star rating> star review for <original
    review business name>. Make the review around <original review length>
    words long.
```

Another example may be for a student essay. We sometimes use double prompts, such as the following:

```
[Prompt] What is a good title for this essay? <original essay> Only give
    the title in your response.

[Assistant] <Title>

[Prompt] Write an essay with the following title: <Title>. Make the essay
    around <original essay length> words long.
```

Another goal of mirror prompting is to increase robustness. A simple way to do this is to prompt the LLM to complete a document, starting with an excerpt from the document. This strongly steers the LLM to write in a style matching the original document, which makes the task harder and improves our learning signal.

An example mirror prompt in this style is:

```
[Prompt] Write a Wikipedia article about <topic>. Start with these
    sentences: <excerpt>. Make the article about <original article length>
    long.
```

Mirror prompts must be hand-tuned for each domain to remove obvious AI tells. For example, when asking for an essay, LLMs often include as the first line a title or "Sure, here is an essay about <topic>: ". Removing these artifacts is vital, otherwise the classifier will simply learn to look only for these tells. This is often done by appending an instruction to the mirror prompt:

```
Do not include a title, word count, or any information besides that of the
    actual essay.
```

In addition to adding this instruction to the prompt side, we also have postprocessing filters that look for the most obvious tells and discard common patterns when found.
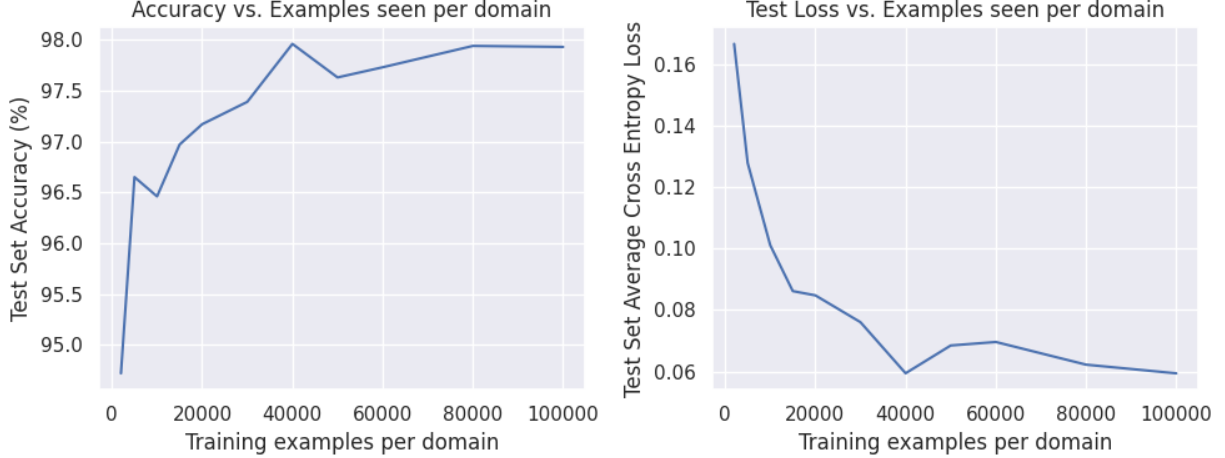
4.3 Scaling Laws

Figure 7: Scaling laws: loss and accuracy against number of training examples seen. Performance begins to saturate around 40,000 examples per domain.

As an initial experiment, we train several classifiers on a series of increasingly large, randomly sampled training sets. Checkpoints are selected based on validation loss and then test loss and test accuracy for each dataset size are computed using that checkpoint. We show in Figure 7 that performance reaches saturation around 40,000 examples per domain (including both real examples and mirrors), which is equivalent to around 400,000 examples total. Even at 100,000 examples per domain, which is 1 million total examples, after saturation has been reached, we see that the model cannot converge to an accuracy above 98%.

4.4 Hard Negative Mining with Synthetic Mirrors

In order to overcome the increasing inefficiency of training a neural network on an extremely large scale dataset of 28 million examples, we developed a hard negative mining algorithm, described below in Algorithm 1. We alternate training a series of neural networks to convergence, with rounds of hard negative mining and synthetic mirror data generation in between in order to surface difficult examples that the model can more efficiently learn from.

Algorithm 1 Hard Negative Mining with Synthetic Mirrors

Input: Large training pool of human examples $\mathbb{P}$, initial training set size n, round size m.
    Initialize training set $T_0 \leftarrow n$ examples randomly sampled from $\mathbb{P}$.
    For each example in T_0, generate mirrored synthetic examples and add them to T_0.
    Train initial model M_0 on T_0 to saturation.
    $i \leftarrow 0$
    while M_i is better than M_{i-1} **do**
        Generate predictions for each example in $\mathbb{P}$ using current model M_i.
        Randomly sample m false positives F_i from predictions.
        For each example in F_i, generate mirrored synthetic examples S_i.
        Create a new training set $T_{i+1} \leftarrow T_i \cup F_i \cup S_i$.
        Train model M_{i+1} on the updated training set T_{i+1}.
        $i \leftarrow i + 1$
    end while

We follow Algorithm 1, hard negative mining with synthetic mirrors, to train our model. Using our scaling laws experiment to inform our choice of initial training pool, we begin with a training set size of 40,000 examples per domain ($n = 360,000$ total), the smallest number required to reach saturation.

For each domain (except student writing - not enough examples), we select up to 10,000 false positives ($m = 80,000$), generate an equal number of synthetic mirrors, add them to the training set and retrain the model.

To track our performance accurately even at very low false positive rates, we hold out approximately 4,000,000 examples from our training pool. These examples are excluded from hard negative mining rounds in order to evaluate the model at each step. The performance of the model on the holdout set implies that hard negative mining surfaces difficult examples that our model can use to generalize, and is not just memorizing hard examples until there are no more remaining.

The results in Table 5 show that this process improves false positive rate across domains, and also results in better performance on email, a domain unseen by either model. The fact that performance on student writing does not change, is indicative that our model is limited by a lack of large-scale training data in this domain and likely has room for improvement in the future.

Interestingly as well, after hard negative mining on other domains, we find that the performance on email (which we intentionally exclude from the training set to demonstrate out-of-domain performance) significantly improves. This suggests that the other domains have a symbiotic relationship with the email domain– adding examples from other domains helps email as well. Therefore, we can expect that adding more domains and scaling up the size our initial training pool even further has the potential to not only cover more domains of text but also improve the model's overall generalization ability.

5 Related Work

5.1 Hard Negative Mining

Hard negative mining has long been used as a technique for object detection in computer vision, as the problem of detecting semantically meaningful objects in images is severely class imbalanced (images are mostly background and therefore most patches are too easy to classify). (Felzenszwalb et al., 2010) (Shrivastava et al., 2016). Our work is related to this problem although we are operating in the language domain rather than the image domain, because most documents are quite easy to classify. Human documents, such as reviews, are often all very similar to each other, and AI documents often have dead giveaways (such as overusing phrases like "delve" and "it is important to note" or "as an AI language model") and so once the model learns from a few examples that a *category* of text is likely to be human or AI, it does not need to see more examples from the category.

5.2 Deep Active Learning

Deep active learning is also a well-researched field of study and an excellent survey can be found here (Ren et al., 2020). Active learning is used when the learning algorithm can interactively query an information source to label new data points. A salient example of active learning being used in a detection task is in (Aghdam et al., 2019) where the neural network's mispredictions of bounding boxes in an object detection task is used to query human labelers for frame-level annotations. While we do not use the learning algorithm to query LLMs to give us new labels, our work is

Domain	Base Model	After Hard Negative Mining
Creative Writing	1.51%	0.19%
Reviews	1.81%	0.09%
Books	0.85%	0.001%
Scientific Papers	1.54%	0.21%
Wikipedia	5.34%	0.23%
News	0.55%	0.04%
Q&A	2.52%	0.13%
ESL	1.44%	0.05%
Student Writing (No Hard Negative Mining)	0.75%	0.75%
Email (Not in training set)	6.60%	0.80%
Total domain-weighted FPR	2.29%	0.25%

Table 5: False positive rate on held out hard negative mining sets, by domain. Hard negative mining reduces false positive rates by 100x-1000x on holdout sets.

related to active learning because we use the underlying neural network to *prompt* the LLMs (the information source) for novel synthetic examples, which guides and updates our training set towards balance that promotes well-conditioned stochastic optimization. Active learning is often used to modify the "working set" of training examples that a classifier such as an SVM or a neural network is trained on. We use our active prompting scheme to augment and improve our working set of examples.

5.3 Probability-Based Feature Classifiers

Probability-based Feature Classifiers are based on the assumption that text generated by an LLM is generally more likely under the model's probability distribution than similar text written by a human. GLTR (Gehrmann et al., 2019) uses entropy, probability, and probability rank to determine whether or not the statistical patterns in the text are AI-generated. The original version of GPTZero (Tian and Cui, 2023) was based on both perplexity and burstiness: the variation of perplexity across the token sequence. DetectGPT (Mitchell et al., 2023) uses perturbations made by a T5 model to compare the probability of a source text to perturbed texts, finding that source texts with a high probability discrepancy against their perturbations are likely to be LLM generated. DetectGPT is a white-box method, meaning that the original LLM used to generate the text is known. However, it was recently shown that smaller language models can be used as proxy models for DetectGPT with reasonable accuracy (Mireshghallah et al., 2023). DetectLLM (Su et al., 2023) builds upon DetectGPT by also introducing similar methods based on log-rank information. Sniffer (Li et al., 2023) uses perplexity-based features to detect AI-generated text and also which LLM produced it.

5.4 Deep Learning Based Classifiers

Several deep learning based classifiers have also emerged as alternatives to probability based classifiers. OpenAI's classifier (OpenAI, 2023b) is a RoBERTa based deep learning model finetuned to predict whether a document is AI-written or not based on a large training set composed of both open-source pretraining data and AI-written data generated by OpenAI's API. Due to its low rate of accuracy, it was discontinued in 2023. More recently, SeqXGPT (Wang et al., 2023) is an example of a hybrid classifier that uses both probabilistic and deep learning elements to show strong performance on sentence level AI classification. It uses extracted perplexities from a variety of language models, and then runs them through a relatively shallow neural network to predict whether sentences are ChatGPT-generated or not. Ghostbuster (Verma et al., 2023) is another recent work that trains a shallow classifier to detect AI-generated text based on combinations of unigram, trigram, and GPT-3.5 embedding features. A recent study (Pu et al., 2023) has shown that deep learning based AI-generated text classifiers exhibit some generalization ability to detect text from larger models when trained on outputs of smaller models from the same family. Finally, RADAR (Hu et al., 2023) uses an adversarial learning approach to increase the robustness of a deep learning based AI detector to paraphrasing attacks. While showing significant progress in detection robustness, like DetectGPT, it is a white-box method, meaning a different detector must be used for each individual LLM, limiting its effectiveness in practice.

5.5 Watermarking

Watermarking is another proposed framework to mitigate harmful usage of AI-generated text. In this setting, the LLM itself leaves behind a probabilistic signal that makes it significantly easier to detect. The watermark proposed by Kirchenbauer et. al (Kirchenbauer et al., 2023) introduces the idea of "green tokens": using some tokens at much higher frequencies than their expectation, allowing a detector with knowledge of the green token set to count the frequency of green tokens and evaluate it against random chance. While watermarking LLMs is an important research topic, with several unwatermarked LLMs already open source and in the wild, the ability to regulate and enforce watermarking regulations as of the time of publication is extremely limited, making a detection solution based on watermarking currently impractical. Furthermore, recent studies (Sadasivan et al., 2023) (Krishna et al., 2023) have shown that watermarking schemes can easily be bypassed using another model that can paraphrase the output of an LLM.

6 Conclusion

In this technical report we have detailed the performance of the Checkfor.ai text classifier against other commercial AI detection systems. We have shown that a simple algorithm for hard negative mining combined with a synthetic mirroring strategy is necessary to scale the performance of a deep-learning based classifier to industry-ready levels of accuracy. We show that the algorithm's novel ability to scale to web-sized datasets enables generalization to out-of-domain text, LLM-generated text where the LLM has not been observed during training, and ability to detect AI-generated text from even the current most capable model, GPT-4.

In future work, we will continue our efforts to increase the robustness of our classifier to adversarial attacks, scale our datasets to even larger sizes, and improve the architecture and training algorithm to further improve the accuracy and generalization ability of the model.

7   Ethics and Responsible Use

Our classifier is available for public commercial use at `https://checkfor.ai`. We strongly discourage the use of our classifier as a sole arbiter of academic integrity and plagiarism checking. All AI detection tools have a nonzero false positive rate, and should be used in conjunction with other evidence to prove or disprove plagiarism.

AI detection is not a substitute nor a reliable tool for proving the factuality or verity of textual information such as news and media. While AI is often used for disinformation, scams, or other malicious activities, human-authored text can also be used with malintent. One should always check the sources of any text in publications for truth independently.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

OpenAI. Chatgpt. `https://chat.openai.com`, 2023a. Large language model.

Google. Gemini: A family of highly capable multimodal models, 2023.

J. Elliott Casal and Matt Kessler. Can linguists distinguish between chatgpt/ai and human writing?: A study of research ethics and academic publishing. *Research Methods in Applied Linguistics*, 2(3), 2023.

Edward Tian and Alexander Cui. Gptzero: Towards detection of ai-generated text using zero-shot and supervised methods, 2023. URL `https://gptzero.me`.

Debora Weber-Wulff, Alla Anohina-Naumeca, Sonja Bjelobaba, Tomáš Foltýnek, Jean Guerrero-Dib, Olumide Popoola, Petr Šigut, and Lorna Waddington. Testing of detection tools for ai-generated text. *International Journal for Educational Integrity*, 19(1), 2023. ISSN 1833-2595. doi:10.1007/s40979-023-00146-z. URL `http://dx.doi.org/10.1007/s40979-023-00146-z`.

Vanderbilt. Guidance on ai detection and why we're disabling turnitin's ai detector, 8 2023. URL `https://www.vanderbilt.edu/brightspace/2023/08/16/guidance-on-ai-detection-and-why-were-disabling-turnitins-ai-detector/`.

Weixin Liang, Mert Yuksekgonul, Yining Mao, Eric Wu, and James Zou. Gpt detectors are biased against non-native english writers, 2023.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. Detectgpt: Zero-shot machine-generated text detection using probability curvature, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

GlobalData. Chatgpt dominates llm (large language models) mentions on social media in 2023, bard distant second. `https://www.hotelnewsresource.com/article127898.html`, 2023. Accessed: 2024-02-08.

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

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mixtral of experts, 2024.

Anthropic. Introducing claude, 2023. URL https://www.anthropic.com/index/introducing-claude.

Fatemehsadat Mireshghallah, Justus Mattern, Sicun Gao, Reza Shokri, and Taylor Berg-Kirkpatrick. Smaller language models are better black-box machine-generated text detectors, 2023.

Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation, 2018.

Vivek Verma, Eve Fleisig, Nicholas Tomlin, and Dan Klein. Ghostbuster: Detecting text ghostwritten by large language models, 2023.

Enron Corp and William W. Cohen. Enron email dataset. https://www.cs.cmu.edu/~./enron/, 2015. Compiled by the United States Federal Energy Regulatory Commission. Distributed by William W. Cohen, Machine Learning Department, CMU. Retrieved from the Library of Congress, https://www.loc.gov/item/2018487913/.

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

Asma and Dina Demner-Fushman. A question-entailment approach to question answering. *BMC Bioinform.*, 20(1):511:1–511:23, 2019. URL https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-019-3119-4.

Fiqa - 2018. https://sites.google.com/view/fiqa/home, 2018. Accessed: 2024-02-13.

Vanderbilt University, The Learning Agency, and Georgia State University. A large-scale corpus for assessing written argumentation: Persuade 2.0. *Zenodo*, 8 2023. URL https://zenodo.org/records/8221504.

Megha Chakraborty, S. M Towhidul Islam Tonmoy, S M Mehedi Zaman, Krish Sharma, Niyar R Barman, Chandan Gupta, Shreya Gautam, Tanay Kumar, Vinija Jain, Aman Chadha, Amit P. Sheth, and Amitava Das. Counter turing test ct2: Ai-generated text detection is not as easy as you may think – introducing ai detectability index, 2023.

Edward Tian. Esl bias in ai detection is an outdated narrative. https://gptzero.me/news/esl-and-ai-detection, 2023.

P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Xiaojiang Chen, and Xin Wang. A survey of deep active learning. *CoRR*, abs/2009.00236, 2020. URL https://arxiv.org/abs/2009.00236.

Hamed Habibi Aghdam et al. Active learning for deep detection neural networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander M. Rush. Gltr: Statistical detection and visualization of generated text, 2019.

Jinyan Su, Terry Yue Zhuo, Di Wang, and Preslav Nakov. Detectllm: Leveraging log rank information for zero-shot detection of machine-generated text, 2023.

Linyang Li, Pengyu Wang, Ke Ren, Tianxiang Sun, and Xipeng Qiu. Origin tracing and detecting of llms, 2023.

OpenAI. New ai classifier for indicating ai-written text. https://openai.com/blog/new-ai-classifier-for-indicating-ai-written-text, December 2023b. Accessed: 2024-01-07.

Pengyu Wang, Linyang Li, Ke Ren, Botian Jiang, Dong Zhang, and Xipeng Qiu. Seqxgpt: Sentence-level ai-generated text detection, 2023.

Xiao Pu, Jingyu Zhang, Xiaochuang Han, Yulia Tsvetkov, and Tianxing He. On the zero-shot generalization of machine-generated text detectors, 2023.

Xiaomeng Hu, Pin-Yu Chen, and Tsung-Yi Ho. Radar: Robust ai-text detection via adversarial learning, 2023.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models, 2023.

Vinu Sankar Sadasivan, Aounon Kumar, Sriram Balasubramanian, Wenxiao Wang, and Soheil Feizi. Can ai-generated text be reliably detected?, 2023.

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense, 2023.