FEW-SHOT DETECTION OF MACHINE-GENERATED TEXT USING STYLE REPRESENTATIONS

Rafael Rivera Soto[1,3,†], **Kailin Koch**[1], **Aleem Khan**[3],
Barry Chen[1], **Marcus Bishop**[2], **Nicholas Andrews**[3,†]
[1]Lawrence Livermore National Laboratory [2]Department of Defense [3] Johns Hopkins University

ABSTRACT

The advent of instruction-tuned language models that convincingly mimic human writing poses a significant risk of abuse. For example, such models could be used for plagiarism, disinformation, spam, or phishing. However, such abuse may be counteracted with the ability to detect whether a piece of text was composed by a language model rather than a human. Some previous approaches to this problem have relied on supervised methods trained on corpora of confirmed human and machine-written documents. Unfortunately, model under-specification poses an unavoidable challenge for neural network-based detectors, making them brittle in the face of data shifts, such as the release of further language models producing still more fluent text than the models used to train the detectors. Other previous approaches require access to the models that may have generated a document in question at inference or detection time, which is often impractical. In light of these challenges, we pursue a fundamentally different approach not relying on samples from language models of concern at training time. Instead, we propose to leverage representations of writing style estimated from human-authored text. Indeed, we find that features effective at distinguishing among human authors are also effective at distinguishing human from machine authors, including state of the art large language models like Llama 2, ChatGPT, and GPT-4. Furthermore, given a handful of examples composed by each of several specific language models of interest, our approach affords the ability to predict which model generated a given document.

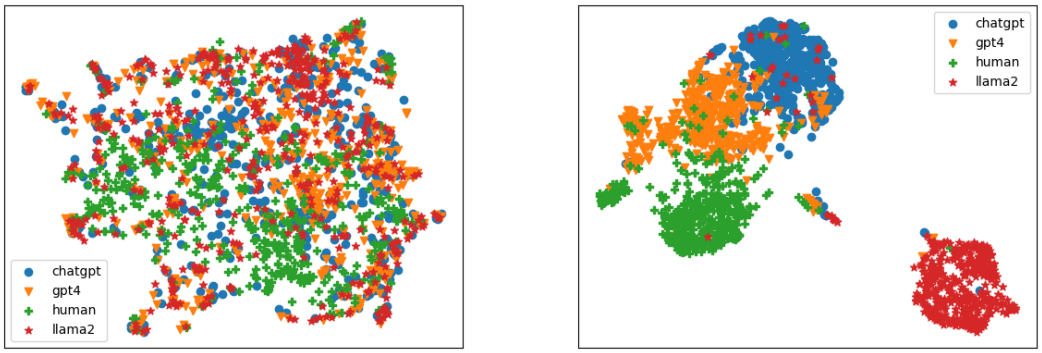
1 INTRODUCTION

Recent interest in large language models (LLM) has resulted in an explosion of LLM usage by a wide variety of users. Some users may regard LLM as something of a writing assistant. Well-intentioned as this use case may be, one concern is that text generated by LLM is not guaranteed to be factually accurate. Therefore such usage may contribute to the spread of misinformation or the reinforcement of biases if documents generated by these models are not carefully edited for accuracy. Other users may intend to use LLM for deception, such as for phishing attacks, disinformation, spam, and plagiarism (Hazell, 2023; Weidinger et al., 2022). Such malicious usage is the chief concern in this work.

To minimize the abuse of *commercial* systems, one recently proposed recourse is for those systems to apply statistical watermarking techniques (Kirchenbauer et al., 2023). However, the advent of open-source LLM with performance approaching that of commercial LLMs and achievable on commodity hardware (Touvron et al., 2023) raises the possibility of circumventing the watermarking mechanisms of commercial systems to generate harmful content, potentially at a large scale. Thus, while watermarking may help mitigate some unintended consequences of LLM adoption, the approach fails to completely address the issue of malicious content.

Because it seems inevitable that people will be exposed to machine-generated text masquerading as human-generated, one reasonable recourse is to develop models to predict whether a given document was composed by a LLM rather than a human. To this end, a number of methods have been proposed, such as OpenAI's machine-generated text classifier (Solaiman et al., 2019). An

[†]Corresponding authors: rafaelriverasoto@jhu.edu, noa@jhu.edu



(a) Semantic document embeddings

(b) Stylistic document embeddings

Figure 1: UMAP projections (McInnes et al., 2018) of semantic or stylistic representations of writing samples in the Reddit domain composed by human or machine authors. We use SBERT as a representative dense semantic embedding (Reimers and Gurevych, 2019) and UAR as a representative stylistic representation (Rivera-Soto et al., 2021). Each point shown is the result of embedding a document with no more than 128 subword tokens for a standard vocabulary of size 50K. Despite using prompts designed to elicit a variety of writing styles from the LLM as discussed in §4.1, the stylistic representation separates humans from machines and machines from one another significantly better than the semantic representation.

obvious drawback of classification methods is that these models require a considerable number of examples generated by LLM and will therefore need to be updated whenever new LLM are introduced, something which could be both impractical and expensive in light of the frequent release of fine-tuned open-source variants. Indeed, prior work in machine-text detection has found that training such a classifier on documents generated by the specific LLM one wishes to detect is necessary to achieve good performance (Zellers et al., 2019). Unfortunately, it is not always the case that sufficiently many documents generated by the LLM one hopes to detect are amply available at training time. Indeed, a detector should ideally perform well when evaluated on documents composed by LLM not contributing to the corpus used to train it, as well as on documents from new domains or dealing with new topics.

On the other hand, we observe that LLM exhibit consistent writing styles across a wide range of prompts, even when explicitly prompted to generate text in specific styles. In order to assess writing style, we propose to employ representations of style learned from large corpora of writing samples by human authors, which aim to capture invariant features of authorship (Wegmann et al., 2022; Rivera-Soto et al., 2021). Figure 1 illustrates that such representations do indeed separate documents composed by humans from those composed by LLM. Moreover, under the additional hypothesis that text generated by any particular LLM follows a style distinct from the styles of human authors and also distinct from those of other LLM, we hope to detect not only that a given document was composed by a machine, but also to classify the LLM that generated it, something that may shed light on the question of which LLM are being abused, increasing transparency and accountability for companies disseminating LLM without appropriate controls or safeguards.

In light of the unavoidable distribution shifts stemming from the introduction of new LLMs, topics, and domains, this work focuses on the few-shot setting. Specifically, our evaluations assess the ability to detect writing samples produced by LLM unseen at training time, and in some cases in novel domains and dealing with novel topics. Finally, given only a few examples of documents composed by several LLM of concern, the proposed approach could be used to classify a machine-generated document according to the LLM that generated it. Our approach differs significantly from prior work in that we do not require access to the predictive distribution of the unseen LLM, like Mitchell et al. (2023), or a large number of samples from it, like Zellers et al. (2019), to effectively detect text generated by these models.

Primary contributions We find that text sampled from LLM like GPT-4 can be reliably detected using style representations and a few short documents generated by LLM of concern. Our approach outperforms dedicated few-shot learning methods as well as standard zero-shot baselines. We also

explore factors leading to effective style representations for this task, finding that contrastive training on large amounts of human-authored text is sufficient to obtain useful representations, but that in certain few-shot settings training on additional LLM-generated documents significantly improves performance. In addition, we release the datasets we generated as part of this work, which include documents generated by a variety of language models.

2 Related Work

Perhaps the most widely applied machine-text detector is OpenAI's AI Classifier, a binary classifier intended to predict whether a given document was human- or machine-generated (Solaiman et al., 2019). The model was trained using documents generated by GPT-2 serving as positive examples and documents drawn from the corpus used to train GPT-2 serving as negative examples. OpenAI released a similar classifier for detecting ChatGPT in January 2023. As of this writing, OpenAI has withdrawn the AI Classifier, citing its "low rate of accuracy." Indeed, it is well known that supervised detectors may overfit various properties of their training data, such as specific decoding strategies (Ippolito et al., 2020).

Perturbing the output logits of a LLM and thereby its decoded text, a recent proposal known as *watermarking*, enables accurate detection of text generated by LLM in some settings. For example, Kirchenbauer et al. (2023) encode a watermark in generated text by splitting a model's vocabulary into so-called *red* and *green* lists, encouraging tokens from the green list to be sampled during decoding. Because this line of work requires direct access to a model and its vocabulary, the approach is most relevant for models deployed by an organization through an API (He et al., 2022). However, adversaries may simply decline to produce watermarked text, for example by using their own LLM, or by removing the watermark from API-generated text. For example, paraphrasing has emerged as an effective mechanism to circumvent detection of a watermarked LLM (Krishna et al., 2023).[1]

Much of the recent work in detecting machine-generated text in a *zero-shot* setting has focused on directly training a classifier using a dataset of human and machine-generated text (Jawahar et al., 2020), requiring access to a model's predictive distribution for comparison with other distributions (Mitchell et al., 2023), or directly using a model to detect its own outputs (Zellers et al., 2019). Other work has looked at patterns of repetition typical of machine-generated detect to rank documents in a corpus likely to be machine-generated (Gallé et al., 2021). In other recent work, the reliability of machine-text detectors in general has been questioned on theoretical grounds (Sadasivan et al., 2023). Zero-shot detection of machine-generated text is a difficult task for human discriminators as well. Indeed, Dugan et al. (2020) demonstrated that human annotators had difficulty pinpointing changepoints between passages of documents composed by humans and those composed by machines, and Maronikolakis et al. (2020) find that automatic detectors can outperform humans in certain settings.

Given the limitations of discriminative classifiers at detecting machine-written documents, one might wonder whether generative approaches would be more robust. Unfortunately, deep generative models are also vulnerable to distribution shifts in general (Nalisnick et al., 2018), whereas effective machine-generated text detectors must be robust to shifts in topic, domain, and sampling techniques relative to those employed at training time. We address this concern by using learned stylistic representations which are trained to ignore features that evolve over time, such as topic and domain, and focus on stylistic features that authors use more consistently (Rivera-Soto et al., 2021; Wegmann et al., 2022).

3 Methods

3.1 Detection regimes

The experiments in this paper deal with both the supervised and few-shot learning regimes. In the supervised setting, we avail of a training corpus consisting of human- and machine-written documents. The goal is to estimate the probability that a held-out document was generated by a LLM. In contrast with the few-shot setting described below, here we assume that we have at least hundreds but preferably thousands or more examples composed by a variety of language models.

[1]We conduct a study comparing our approach to watermarking in Appendix G.

All being well, the classifier would generalize to various test conditions, including novel language models, topics, and domains. In practice the probability estimate is implemented by fine-tuning an underlying, pre-trained feature representation. We consider two such representations in this paper, namely *semantic representations*, where the features are obtained by fine-tuning pre-trained representations based on masked language models, and *style representations*, where the features are obtained from pre-trained style representations, which we discuss in more detail in §3.2. Results on the generalization of supervised detectors are reported in Appendix A.

In the few-shot setting we assume that for each LLM of concern we have a *support sample*, which consists of a small number of *in-distribution* documents composed by that model. For example, upon recognizing by manual inspection that some of her students' assignments exhibit telltale signs of machine-generation, such as hallucination, an instructor could take the support sample to comprise those essays in question. Alternatively, the instructor might proactively arrive at a support sample by prompting various LLM herself. The goal in this setting is to estimate the probability that one of the LLM in question generated a given document, and if so, a secondary objective is to predict which model generated the document.

3.2 Style representations for machine-written text detection

A primary component of our proposed approach to detecting machine-generated text is the notion of a *style representation*, something we introduce to help our detectors overcome the challenges of generalization to new LLM, topics, and domains. More formally, we denote a style representation below generically by f, which is taken to be some auxiliary model. The representation f maps a handful documents $x_1, x_2, \ldots, x_K$ to a fixed-dimensional vector $f(x_1, x_2, \ldots, x_K)$, which is also known as a *representation* of the document collection. The mapping f is fit such that $f(x_1, x_2, \ldots, x_K)$ and $f(x'_1, x'_2, \ldots, x'_L)$ have large cosine similarity if and only if the style of $x_1, x_2, \ldots, x_K$ is similar to that of $x'_1, x'_2, \ldots, x'_L$.

An important observation is that writing style comes into focus only after observing a sufficiently large writing sample. For example, the repeated usage of a rare word may be discriminative of a particular author, but observing repeated word usage typically requires observing more than a few sentences. Indeed, our best results are obtained by providing models with longer spans of text by aggregating style representations of multiple short documents.

In light of the highly nuanced nature of writing style, learning generalizable style representations requires large amounts of data. To this end, prior work has leveraged the availability of proxy author labels available in the form of account names on various social media platforms, such as blogs, microblogs, technical fora, and product reviews. In this paper, we focus on Reddit posts, which provide writing samples from authors discussing a wide variety of topics in various communities, which are known as *subreddits*. Furthermore, these topics cover diverse author interests and backgrounds, resulting in a corpus representing diverse styles. Our datasets are described in more detail in §4.1.

Training style representations The primary challenge in learning style representations is in disentangling time-invariant features, chiefly writing style, from time-varying features, such as topic. To achieve this, we employ two contrastive training strategies, depending on available data. First, if a dataset includes histories of each author's writing over several months or years, then we pair writing samples composed at different points in time by the same author to yield *positive examples*, and we pair writing samples by different authors to yield *negative examples* (Andrews and Bishop, 2019). Alternatively, if topic labels corresponding with each writing sample are available, then we construct *hard positives* by pairing writing samples composed by a single author discussing different topics, and *hard negatives* by pairing samples composed by different authors discussing similar topics (Wegmann et al., 2022). Besides the use of hard negative mining, the approach of Wegmann et al. (2022) differs in that non-episodic training is used; that is, features are computed on individual documents rather than episodes of multiple documents. In addition to style representations trained using both strategies, we also use a non-stylistic representation as a baseline, namely SBERT (Reimers and Gurevych, 2019).

Proposed few-shot method An author-specific style representation f admits a straightforward mechanism for few-shot detection. Specifically, suppose $x_1, x_2, \ldots, x_K$ is a handful of documents

known to have been generated by a particular language model of interest, where we regard a *document* to be short span of text of around the length of a social media post. Specifically, most documents considered in this work have length around 128 tokens according to a fixed tokenizer. At inference time, given a handful of new of documents $x'_1, x'_2, \ldots, x'_L$, we compute the cosine similarity between $f(x_1, x_2, \ldots, x_K)$ and $f(x'_1, x'_2, \ldots, x'_L)$ to obtain a score monotonically related to the estimated probability that $x'_1, x'_2, \ldots, x'_L$ were composed by the same target language model as $x_1, x_2, \ldots, x_K$. The score may be further calibrated to yield a meaningful confidence estimate, which we describe in Appendix D.

In our experiments, we focus on the UAR model, a RoBERTa-based architecture trained with a supervised contrastive objective according to the recipe described in Rivera-Soto et al. (2021) and use their reference implementation. We observed in preliminary experiments that increasing the number of human authors contributing to the dataset used to train UAR improves performance on the authorship attribution task and improves generalization. Therefore, we train UAR using a larger dataset than in prior work, namely a corpus of Reddit comments composed by five million authors as described in §6. In addition to this instance of UAR, we also experiment with the following variations.

Multi-domain variation Previous work has shown that including multiple domains during training can improve the quality of style representations (Rivera-Soto et al., 2021). To this end, we arrive at our first variation of the representation above by augmenting the training dataset with data drawn from Twitter and StackExchange. When training UAR with this augmented dataset, we make a small modification to the training procedure. Namely, we randomly sample the domain of each example before sampling the example itself from that domain. See Table 4 for further statistics of this augmented dataset.

Multi-LLM variation In our second variation, we initialize the model weights of UAR according to the instance above trained on the comments of five million Reddit users. We continue training on posts by human authors drawn from `r/politics` and `r/PoliticalDiscussion` as well as posts to these subreddits generated by LLM, as discussed in §4.1. By restricting to these subreddits, we aim to control for topic, thereby introducing an inductive bias encouraging the representation to separate machine and human authors on the basis of features unrelated to topic. In this setting we regard a language model as an author distinct from other members of its LLM family. For example, we regard GPT2-large as a distinct from GPT2-xl. We also ensure that the lengths of the posts generated by LLM match those of the human-generated posts by truncating each sample along sentence boundaries when necessary. Controlling for topic and length ensures that the model avoids learning non-stylistic shortcuts that would simplify the task.

4 EXPERIMENTS

4.1 DATASETS

In the following experiments we distinguish between two kinds of language models, namely amply available and cheap (AAC) models, and models that users are likely to want to detect (LWD). We assume that documents generated from AAC models are available at training time, when the feature representations f are estimated or fine-tuned. On the other hand, documents generated by LWD models are held-out for evaluation, simulating the emergence of new LLM with more powerful capabilities, including the ability to better mimic human authorship.

Training corpus To create the AAC dataset, we generated data using two variants of easily available models, namely GPT-2 (Radford et al., 2019) and OPT (Zhang et al., 2022). For GPT-2 we use the "large" and "XL" variants, which have 774M and 1.5B parameters respectively. The OPT models have 6.7B and 13B parameters. We generated text using human-generated posts to `r/politics` and `r/PoliticalDiscussion` as input prompts. All prompts contain at least 64 tokens according to the GPT-2 tokenizer. Completions were generated using a range of decoding strategies and values. See Table 6 for these parameters. In total, there are 64 variations of decoding strategies, values and model size. After generating the completions, we truncate each document along sentence boundaries and ensure that all documents contain as close to 64 tokens as possible. Controlling for topic and length is intended to ensure that models trained with this corpus can't easily distinguish between human and machine text by learning extraneous features, such as topic and length.

Evaluation corpora We evaluate our proposed detection approaches using the LWD dataset that we now describe, as well as M4 (Wang et al., 2023), a recently released dataset containing the contributions of multiple LLM in five domains. We construct the LWD dataset by generating text with Llama2, GPT4, and ChatGPT using prompts of the form `write an Amazon Review in the style of the author of the following review: ⟨human review⟩`, where `⟨human review⟩` is a real Amazon Review. Prior work has shown that LLM have some ability to reproduce styles provided as in-context examples (Reif et al., 2022; Patel et al., 2023). Thus, our data generation procedure aims to induce stylistic variety to the extent currently possibly by state of the art LLM, with the goal of increasing the challenge of our benchmark. More details on the LWD and M4 datasets can be found in Appendix E.

4.2 Few-shot and zero-shot detection baselines

We compare our proposed few-shot approach to the following baseline models. We use episodic training to fit *prototypical networks* (Snell et al., 2017) and MAML (Finn et al., 2017) using the AAC dataset described in §4.1 together with human-generated posts to `r/politics` and `r/PoliticalDiscussion`, where each distinct LLM contributing to AAC is regarded as a single author. In addition, we perform ablations on our dataset choices in Table 3. Next, we introduce variations of our proposed approach in which we replace the UAR representation with alternative embeddings, namely SBERT (Reimers and Gurevych, 2019)[2] and CISR (Wegmann et al., 2022)[3], a further authorship representation that leverages hard negatives and positives at training time. Additionally, we compare to several zero-shot baselines, including two versions of the OpenAI detector Solaiman et al. (2019), one off-the-shelf and one which we retrain on Reddit politics posts. We further include detectors based on metrics derived from the GPT2-XL likelihood, including `Rank`, `LogRank`, and `Entropy`, proposed in Gehrmann et al. (2019); Solaiman et al. (2019); Ippolito et al. (2020), respectively.

4.3 Metrics

We use the Receiver Operating Characteristic (ROC) curve to assess detection performance as the corresponding detection threshold varies. To summarize the ROC and compare different methods across operating points, we report the standardized partial area under the ROC curve (AUC) restricted to the range of operating points corresponding with false alarm rates not exceeding 1%, which we denote by pAUC in this work. This allows us to better compare high-performing systems, noting that the proposed few-shot learning methods achieve near perfect AUC when calculating the area under the entire ROC curve, even when supplied with very few examples generated by LLM of concern. Our point of view is that users will be most interested in the range of operating points corresponding with low false alarm rates, as these are often most relevant for users, who might mistrust and eventually ignore systems that raise too many false alarms. This is intended to help users arrive at operating points that minimize time-consuming followup inspection. We report further experiments with even smaller writing samples in Appendix H, with additional metrics including AUC and FPR@95.

4.4 Single-target machine text detection

In this section, we assume access to a small *in-distribution* writing sample from a *specific* model of concern, such as ChatGPT. The objective is to identify further writing samples from this specific model appearing in new documents. This evaluation reflects the setting where one would like to perform targeted detection of a particular LLM. For example, an instructor might like to be alerted to cases where their students may have used a particular LLM as a writing assistant. In §4.5 we evaluate the same detection approaches in the setting where the task is to identify documents generated by *any* of *multiple* LLM, noting that the current task is somewhat more difficult because it scores the prediction of a LLM as incorrect if the query was generated by a LLM *different* than the one predicted.

We evaluate all the detection approaches considered on the evaluation corpora described in §4.1. Each corpus contains documents composed by humans and LWD models drawn from a particular text domain other than Reddit, which serves as the primary training domain. All documents in each

[2] `https://huggingface.co/sentence-transformers/paraphrase-distilroberta-base-v1`

[3] `https://huggingface.co/AnnaWegmann/Style-Embedding`

corpus are grouped into *episodes* of N documents by the same human or machine author for some value of N, each document containing a maximum of 128 tokens. These episodes serve as the writing samples to be evaluated for machine-authorship.

We apply the following procedure for each detection approach considered and for each evaluation corpus. For every LLM model contributing to the evaluation corpus, we randomly select one episode to serve as the *support sample* of the LLM that generated it, with all remaining evaluation episodes serving as *queries*. We calculate a score for every query indicating how similar it is to the support sample, noting that the way this score is calculated depends on the detection method being evaluated. Each of these scores is paired with a label indicating whether the query was composed by the same LLM as the support sample, or composed by a human or a different LLM. Finally, we calculate a ROC curve based on these scores and labels. We repeat this process a maximum of 1000 times for each LLM and for each detection approach except MAML, each time randomly sampling a new support sample, and report the average value of pAUC over the 1000 trials. For MAML we repeat the process only 20 times due to its computational burden.

The results are shown in Table 1, which reports the mean and standard error over all support samples and each evaluation corpus for each detection approach considered. The proposed method based on representations of writing style outperforms all other approaches. We note that the RoBERTa ProtoNet detector is trained on the AAC corpus and has 40M more parameters than UAR, so the superior performance of the style representation methods is not a consequence of larger model capacity. We explore the effect of modifying the episode size of the *queries* to $N = 1$ in Table 13. We also break down these results according to evaluation domain in Table 16.

Method	**Training Dataset**	**pAUC** $N = 5$	**pAUC** $N = 10$
Few-Shot Methods			
UAR	Reddit (5M)	**0.905 (0.001)**	**0.9806 (0.0006)**
UAR	Reddit (5M), Twitter, StackExchange	0.886 (0.001)	0.9676 (0.0008)
UAR	AAC, Reddit (politics)	0.877 (0.001)	0.9400 (0.0013)
CISR	Reddit (hard neg/hard pos)	0.839 (0.001)	0.9331 (0.0013)
RoBERTa (ProtoNet)	AAC, Reddit (politics)	0.871 (0.001)	0.9475 (0.0014)
RoBERTa (MAML)	AAC, Reddit (politics)	0.662 (0.006)	0.6854 (0.0068)
SBERT	Multiple	0.621 (0.002)	0.7157 (0.0022)
Zero-Shot Methods			
AI Detector (fine-tuned)	AAC, Reddit (politics)	0.6510 (0.031)	0.6585 (0.0320)
AI Detector	WebText, GPT2-XL	0.6028 (0.0250)	0.6011 (0.0249)
Rank (GPT2-XL)	BookCorpus, WebText	0.5693 (0.0152)	0.5581 (0.0172)
LogRank (GPT2-XL)	BookCorups, WebText	0.7640 (0.0360)	0.7749 (0.0378)
Entropy (GPT2-XL)	BookCorpus, WebText	0.4984 (0.0005)	0.4977 (0.0002)
Random		0.005	0.005

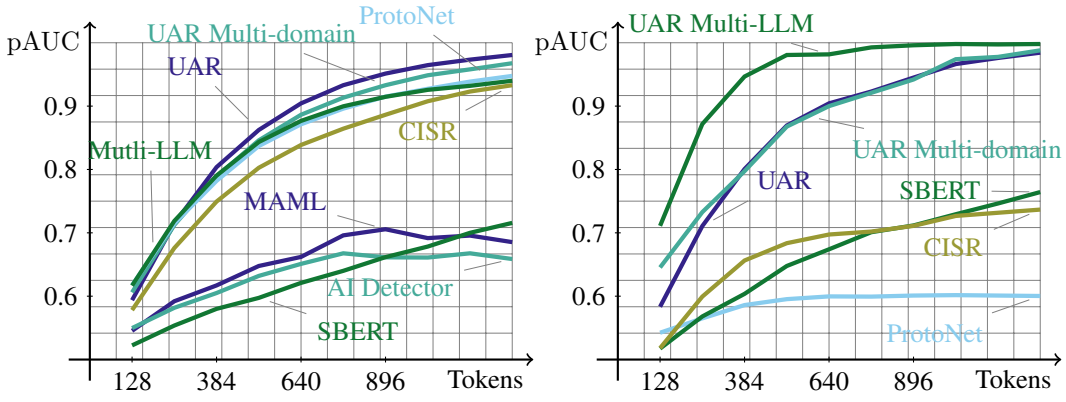
Table 1: Single-target detection results. Each model was evaluated on a common corpus of documents involving of unseen domains, topics, and LLM, organized into episodes of N documents. The standard errors shown in parenthesis were estimated using bootstrapping. The metric pAUC defined in §4.3 is *strictly* less than the usual AUC.

To study the effect of the number N of documents comprising each episode, we vary N between 1 and 10, still truncating each document to the nearest sentence boundary before the hundred-twenty-eighth token. The results are shown in Figure 2a. We observe that UAR trained on Reddit performs best across the range, although other UAR variants perform well also. Furthermore, stylistic representations do not require AAC data during training to capture the style of machine generators. Both UAR and CISR outperform methods that require AAC training data.

We also observe that metric-based approaches like RoBERTa (ProtoNet) outperform fast-adaptation approaches like RoBERTa (MAML). We hypothesize that this is because RoBERTa (MAML) is limited to 512 tokens at inference time, while metric-based approaches may combine representations of various spans of text together, thus effectively increasing the context from which they make predictions. Another reason for this discrepancy might be that fast-adaptation approaches are more prone to overfitting the support sample.

4.5 MULTIPLE-TARGET MACHINE TEXT DETECTION

In §4.4 we handled the case of detecting a *single* target language model. We now extend our formulation to include *multiple* target language models, given support samples from each model. For each detection approach considered we apply the process in §4.4 for each target language model independently to arrive at a score for that model. Then the minimum of these scores is used to predict whether *any* of the target language models generated a given query. As in §4.4 we vary the number N of documents between 1 and 10, still truncating each document to the nearest sentence boundary before the hundred-twenty-eighth token. The results are shown in Figure 2b.



(a) Single-target detection

(b) Multiple-target detection

Figure 2: Detection performance as the number of input tokens varies.

We find that in this setting, the proposed approach with UAR trained on both Reddit and AAC performs the best, although the other UAR variants remain competitive. We note that the evaluation corpus contributing most to the difference is the `PeerRead` component of M4, as shown in Table 17, which breaks down the results of this experiment by evaluation dataset. Otherwise the results are similar for all evaluation domains. We hypothesize the AAC variant is better able to capture the differences between different LLM, whereas the UAR variants trained on only human-generated documents capture the differences between humans and machines more generally. We also find that the ProtoNet and CISR approaches no longer perform well in this setting, indicating that both models may be poor at distinguishing among LLM.

Note that in this experiment, a key assumption is that each support sample is known to have originated *entirely* from one of several LLM of concern. However, it may be possible, say, by manual inspection, to ascertain that each of a handful of documents was generated by *some* LLM, but not necessarily all by the *same* LLM. This setting is addressed in Appendix I.

4.6 ROBUSTNESS AGAINST PARAPHRASING ATTACKS

We now test the robustness of our proposed approach against an adversary that paraphrases the generated text to evade detection. To simulate this scenario, we paraphrase the generated data in the unseen domains using DIPPER (Krishna et al., 2023) with a lexical diversity parameter of 20%. We test two versions of UAR, one where the support examples come strictly from the un-paraphrased LLM, and another that mimics the multiple-target (§4.5) scenario in that we provide UAR with one support example of the un-paraphrased LLM and another of the paraphrased LLM. For both experiments, we vary the proportion of queries that are paraphrased from 0% to 100% and set $N = 5$. The results are shown in Figure 3.

We observe that both UAR and ProtoNet suffer as the proportion of queries paraphrased increases. However, including the paraphrased LLM as a support example (UAR Multi-LLM) ameliorates the drop in performance.

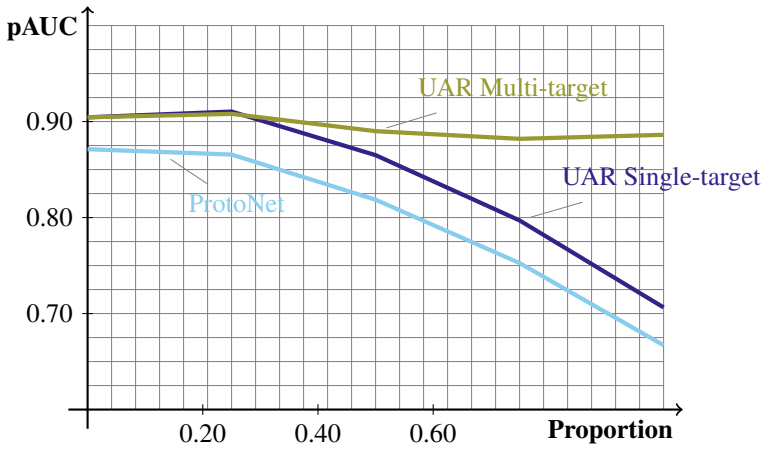


Figure 3: Mean of pAUC as a proportion of the *queries* is paraphrased. Paraphrasing reduces the detection rate across the low FPR range ($\leq 1\%$). Including the paraphrased LLM as a support example (UAR Multi-LLM) ameliorates the drop in performance.

5 Conclusion

Summary of findings We propose a few-shot strategy to detect machine-generated text using style representations estimated from content primarily composed by humans. Our main finding is that style representations afford a remarkable ability to identify instances of text composed by LLM given only a handful of demonstration examples, even when those examples were generated with prompts engineered to elicit diverse writing styles. In addition, we have shown in §4.5 that further improvements are possible for multiple-target LLM detection by fine-tuning style representations on documents generated by AAC models and in §4.6 we've shown that the multiple-target variant of the proposed approach is robust against paraphrasing attacks. By focusing our evaluation on the portion of the ROC curve corresponding to a false alarm rate of less than 1%, we seek to emphasize that the proposed methodology represents a practically relevant approach to mitigating certain LLM abuses.

Limitations The few-shot detection framework explored in this work assumes access to a handful of documents generated by LLM that users may wish to detect in a particular domain. As of this writing, there are only a small but growing number of such LLM, since only a handful of large companies have the resources to train such models. This makes it possible to anticipate which LLMs an adversary may abuse and proactively train detectors using *any* of the approaches considered in this work. However, in the future it seems plausible that a much larger number of LLM will be available. Under such circumstances a more reactive approach would be required, such as the approach proposed in this paper, where examples of abuse by unknown LLM are assembled to form the required support samples.

Our experiments focus on English since LLM of concern are primarily available in English. However, since the training procedure for the style representations used in our few-shot detection experiments relies only on the availability of author-labeled text, there are no barriers to developing such representations for other languages other than collecting sufficiently large corpora. We acknowledge that this is easier accomplished for high-resource languages, particularly those that are well-represented in online discourse. For these reasons we believe exploring style representations effective in low-resource languages is an interesting avenue for future work.

Broader impact The rapid adoption and proliferation of LLM poses a risk of abuse unless methods are developed to detect deceitful writing. The proposed few-shot detection method represents a novel and practical approach to detecting machine-generated text in many settings, including plagiarism in classrooms, social media moderation, and email spam and phishing. The approach may be deployed immediately using readily available, pre-trained style representations and requires only a small number of examples generated by LLM of concern. We will release code and checkpoints of our best models to facilitate adoption of the proposed approach.

6 REPRODUCIBILITY

The proposed few-shot detectors are trained using a reference open-source PyTorch implementation of UAR available at `https://github.com/LLNL/LUAR`, using default hyperparameter choices. For the CISR baseline, we use the open-source PyTorch implementation available at `https://github.com/nlpsoc/Style-Embeddings`. For ProtoNet and MAML, we use the implementations provided in `https://github.com/learnables/learn2learn`. The data used to fine-tune the UAR style representations is sampled from a publicly available corpus of Reddit comments (Baumgartner et al., 2020). We subsampled this dataset for comments published between January 2015 and October 2019 by authors publishing at least 100 comments during that period. Additionally, we use Amazon reviews and StackExchange discussions in some model variations (Ni et al., 2019), both obtained from existing datasets. The Amazon dataset may be downloaded from `https://nijianmo.github.io/amazon/index.html` and the StackExchange dataset is available from `https://pan.webis.de/clef21/pan21-web/style-change-detection.html`. We also create two new corpora of machine-generated documents, referred to as AAC and LWD in the main text, which we use respectively for training and evaluation. In the case of AAC, we use publicly released checkpoints for GPT-2 and OPT, available at the time of this writing from `https://huggingface.co/models`. In the case of LWD, we used the OpenAI API for ChatGPT and GPT-4 generations. We generated Llama2 examples using the Llama2 7B chat model released July 2023, which can be found at `https://github.com/facebookresearch/llama`. Training of the style representations was performed using one 8 x A100-80Gb GPU server, which took under 24 hours for each of the proposed model variations. In the case of both UAR and CISR, the resulting style feature extractors have only 82M and 125M parameters respectively, and are therefore efficient to deploy on a single GPU. All our datasets, along with the code we used to prepare them and the code we used to fine-tune UAR variants and run the various experiments will be released at a later date.

7 ACKOWLEDGEMENTS

Part of this work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DEAC52-07NA27344. This work was also supported by the Human Language Technologies Center of Excellence at Johns Hopkins University and by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the HIATUS Program contract D2022-2205150003. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

REFERENCES

2023. OpenAI ChatGPT API "gpt-3.5-turbo". Available at: `https://api.openai.com/v1/chat/completions`.

Nicholas Andrews and Marcus Bishop. 2019. Learning invariant representations of social media users. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1684–1695.

Jason Baumgartner, Savvas Zannettou, Brian Keegan, Megan Squire, and Jeremy Blackburn. 2020. The Pushshift Reddit Dataset. In *Proceedings of the 14th International AAAI Conference on Web and Social Media (ICWSM)*, volume 14, pages 830–839.

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. 2020. Language models are few-shot learners.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free dolly: Introducing the world's first truly open instruction-tuned llm.

Liam Dugan, Daphne Ippolito, Arun Kirubarajan, and Chris Callison-Burch. 2020. Roft: A tool for evaluating human detection of machine-generated text.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks.

Matthias Gallé, Jos Rozen, Germán Kruszewski, and Hady Elsahar. 2021. Unsupervised and distributional detection of machine-generated text. *arXiv preprint arXiv:2111.02878*.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander M Rush. 2019. Gltr: Statistical detection and visualization of generated text. *arXiv preprint arXiv:1906.04043*.

Tilmann Gneiting and Adrian E Raftery. 2007. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR.

Julian Hazell. 2023. Large language models can be used to effectively scale spear phishing campaigns. *arXiv preprint arXiv:2305.06972*.

Xuanli He, Qiongkai Xu, Yi Zeng, Lingjuan Lyu, Fangzhao Wu, Jiwei Li, and Ruoxi Jia. 2022. Cater: Intellectual property protection on text generation apis via conditional watermarks.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2020. Automatic detection of generated text is easiest when humans are fooled. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1808–1822.

Ganesh Jawahar, Muhammad Abdul-Mageed, and Laks Lakshmanan, V.S. 2020. Automatic detection of machine generated text: A critical survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2296–2309, Barcelona, Spain (Online). International Committee on Computational Linguistics.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2023. A watermark for large language models. *arXiv preprint arXiv:2301.10226*.

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. 2023. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense. *ArXiv*, abs/2303.13408.

Antonis Maronikolakis, Hinrich Schutze, and Mark Stevenson. 2020. Identifying automatically generated headlines using transformers. *arXiv preprint arXiv:2009.13375*.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

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

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, Xiangru Tang, Dragomir Radev, Alham Fikri Aji, Khalid Almubarak, Samuel Albanie, Zaid Alyafeai, Albert Webson, Edward Raff, and Colin Raffel. 2023. Crosslingual generalization through multitask finetuning.

Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. 2018. Do deep generative models know what they don't know? *arXiv preprint arXiv:1810.09136*.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 188–197.

OpenAI. 2023. Gpt-4 technical report.

Ajay Patel, Nicholas Andrews, and Chris Callison-Burch. 2023. Low-resource authorship style transfer: Can non-famous authors be imitated?

John Platt et al. 1999. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Emily Reif, Daphne Ippolito, Ann Yuan, Andy Coenen, Chris Callison-Burch, and Jason Wei. 2022. A recipe for arbitrary text style transfer with large language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 837–848, Dublin, Ireland. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Rafael A. Rivera-Soto, Olivia Elizabeth Miano, Juanita Ordonez, Barry Y. Chen, Aleem Khan, Marcus Bishop, and Nicholas Andrews. 2021. Learning universal authorship representations. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 913–919, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, et al. 2019. Release strategies and the social impacts of language models (`arxiv:1908.09203`). `https://huggingface.co/roberta-base-openai-detector`.

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. 2023. Llama 2: Open foundation and fine-tuned chat models.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Chenxi Whitehouse, Osama Mohammed Afzal, Tarek Mahmoud, Alham Fikri Aji, and Preslav Nakov. 2023. M4: Multi-generator, multi-domain, and multi-lingual black-box machine-generated text detection.

Anna Wegmann, Marijn Schraagen, and Dong Nguyen. 2022. Same author or just same topic? Towards content-independent style representations. In *Proceedings of the 7th Workshop on Representation Learning for NLP*, pages 249–268. Association for Computational Linguistics.

Laura Weidinger, Jonathan Uesato, Maribeth Rauh, Conor Griffin, Po-Sen Huang, John Mellor, Amelia Glaese, Myra Cheng, Borja Balle, Atoosa Kasirzadeh, et al. 2022. Taxonomy of risks posed by language models. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, pages 214–229.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. *Advances in neural information processing systems*, 32.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models.

A GENERALIZATION OF SUPERVISED MACHINE-TEXT DETECTORS

In order to simulate future scenarios, we train various models on text dealing with certain topics, drawn from certain domains, and with positive examples generated by older language models (AAC) and evaluate these models on text dealing with new topics, drawn from new domains, and with positive examples generated by the newest language models. More information on these datasets can be found in Appendix E. In addition, for this experiment we also generated Reddit posts, some examples of which are shown in Table 10. The purpose of the experiment is to determine whether discriminative classifiers built on top of pre-trained (and frozen) style representations are more robust to changes in topic, domain, and language model than those built on top of semantic representations.

To address this question, we train a supervised detector constructed simply by composing an MLP with the frozen UAR Reddit model, pretrained on human-generated text drawn from the AAC political Reddit dataset. Following OpenAI's GPT-2 detector, we also finetune a RoBERTa model. We fit both models with the same amounts of human-generated and AAC text, with each document truncated to the nearest sentence boundary before the sixty-fourth token. See and Table 9 for details on the training and evaluation data.

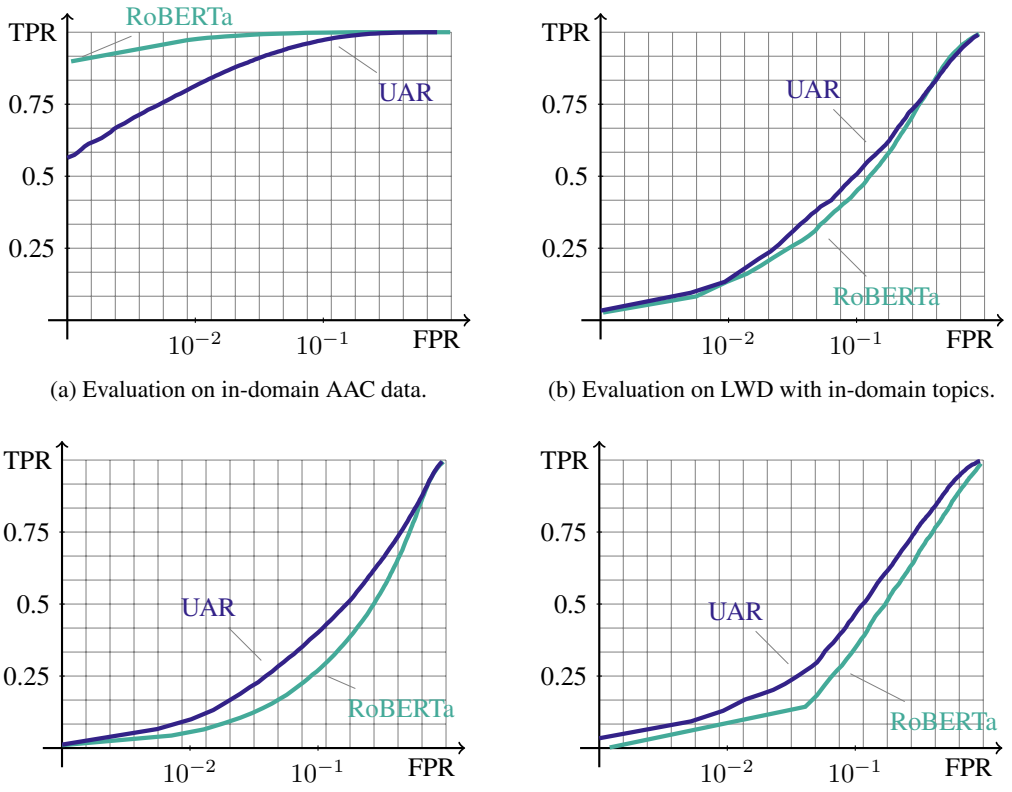
The results are shown in Table 2. In the supervised context, where the detectors have seen both the same topics and language models at training time, both the UAR MLP and the baseline perform strongly. In fact, the RoBERTa baseline outperforms UAR, particularly at the lowest false positive rates, as shown in Figure 4a. However, when the same detectors are evaluated on new topics and domains, performance drops sharply, as shown in Figure 4b, Figure 4c, Figure 4d. In other words, UAR MLP is more robust in the face of these topic and domain shifts than the baseline. However, both the performance of both detectors degrades to the point they would be unreliable as the FPR tends to zero. For example, at a FPR of 1% both models reach TPR of around 40% on LWD datasets. This confirms our expectation about the limitations of using trained classifiers on unseen LWD data.

B PROTONET ABLATIONS

To understand the effect of controlling for topic and length in our AAC training data, we perform a set of ablation experiments measuring the pAUC as we ablate these choices. Below, we can see that if we controlled for topic only, the model is able to learn document length as an extraneous feature by which to distinguish humans from machines, resulting in a relative decrease in pAUC of 9.7%. Further ablating our control of topics doesn't have as large of an effect.

Evaluation Set	UAR	RoBERTa
AAC	0.8551	0.9671
LWD	0.5401	0.5363
LWD, new topics	0.5262	0.5136
LWD, new domains	0.5411	0.5007
Random	0.005	0.005

Table 2: pAUC scores for UAR and RoBERTa baseline on AAC and LWD.



(a) Evaluation on in-domain AAC data.

(b) Evaluation on LWD with in-domain topics.

(c) Evaluated on LWD on out-of-domain topics.

(d) Evaluated on LWD on unseen domains.

Figure 4: Supervised machine-text detection performance. Both the RoBERTa detector and the UAR detector perform well in-distribution, but performance drops when evaluating on LWD, new topics, and new domains. The UAR detector trained on stylistic representations is more robust to changes in the testing distribution.

Ablation	pAUC
Control Topic and Length	0.844 (0.009)
Control Topic Only	0.769 (0.002)
No Control	0.765 (0.002)
Random	0.005

Table 3: Mean of pAUC across on Amazon. Each number is the mean of the distribution along with the standard error, estimated via bootstrapping.

C UAR TRAINING DATASET STATISTICS

Table 4 shows statistics on the number of authors and number of examples in each of the training sets used to estimate the UAR style representations used in our proposed few-shot detection method.

Dataset	Number of Authors	Number of Examples
StackExchange	22,469	2,758,657
Twitter	1,905,705	370,277,856
Reddit	5,199,959	3,578,220,305

Table 4: Dataset Statistics

D CALIBRATION

In certain applications, machine-generated text detectors are most useful when they provide *interpretable* probabilistic scores to enable risk-aware decision making. For this reason we may wish to further calibrate detector scores into probability estimates in both the supervised and few-shot settings. In principle we expect model predictions to be fairly well-calibrated in the supervised setting by virtue of the fact that these models are typically optimized using binary cross-entropy, which is a proper scoring function (Gneiting and Raftery, 2007). However, in practice, scores parameterized by deep neural networks often suffer from overconfidence unless some form of post-hoc calibration is applied (Guo et al., 2017).

We calibrate the UAR few-shot detector scores and measure its calibration against a supervised RoBERTa classifer. The expected calibration error (ECE) for multiple evaluation sets can be found in Table 5. To calibrate the UAR similarity scores and RoBERTa probabilities we use Platt's Scaling Platt et al. (1999). For in-domain, we calibrate on Reddit data from M4 and our generated AAC dataset. For the LWD data, we calibrate on 10 human and 10 machine examples. Evaluation sets include a held-out sets of the Reddit AAC texts (in-domain), LWD Reddit texts and LWD out-of-domain texts. Not surprisingly, the in-domain calibration is best, and more text tends to produce lower calibration error. RoBERTa is particularly poorly calibrated on the LWD models and out-of-domain texts.

Evaluation Set	Max Tokens	UAR	RoBERTa Calibrated	RoBERTa
In Domain	128	0.0648	0.1244	0.1425
	256	0.1163	0.0350	0.0790
	512	0.0941	0.0351	0.0789
LWD + Reddit	128	0.1565	0.1657	0.3976
	256	0.1764	0.0756	0.3896
	512	0.1645	0.0696	0.4090
LWD + New Domains	128	0.1147	0.1149	0.4375
	256	0.0990	0.0283	0.4795
	512	0.0855	0.0162	0.4822

Table 5: Expected calibration error for UAR and RoBERTA baseline.

E FURTHER DETAILS ABOUT SYNTHETIC DATASETS

In this section we elaborate on the process we used to create using AAC and LWD language models. For both datasets, we used a single set of prompts to create a variety of documents. Because this resulted in more machine-generated examples than human-generated, we balanced the each dataset with additional human examples before splitting into training, validation and testing splits.

We generated AAC datasets using with all possible combinations of the parameters specified in Table 6. Some statistics of the AAC datasets are shown in Table 7.

Generation Type	Values
Models	GPT2-large, GPT2-xl, OPT-6.7B, OPT-13B
Decoding Strategy	top-p, typical p
Decoding Values	0.7, 0.95
Temperature Values	0.7, 0.9
Generation Length	512 tokens

Table 6: AAC generation parameters.

AAC LMs	Train	Valid	Test
Machine Generated Texts	440,721	62,935	125,987
Human Written Texts	440,721	62,935	125,987
Total Texts	881,442	125,870	251,974

Table 7: AAC dataset statistics.

We used a similar approach to generate our LWD datasets. The models used were GPT-4, ChatGPT and Llama-2. We prompted each LWD model with an example human text of at least 64 tokens. For documents prompted with Reddit posts, we varied the prompts to elicit a diverse range of writing styles by specifying some personality traits of the supposed author. Some examples are shown in Table 10. For the documents prompted with Amazon and Stack Exchange posts, we prompted the language model to preserve the style of the post. Due to cost, we used only GPT-4 to generate Reddit content, but not Amazon or StackExchange.

Finally, in addition to our LWD dataset, we also evaluate using the recently-released M4 dataset (Wang et al., 2023), which consists of machine-generated documents by multiple language models in multiple domains. The domains include ArXiv, PeerRead, Reddit, WikiHow, and Wikipedia. All domains except Reddit are completely unseen for all methods considered in this work, for a total of six unseen domains. M4 includes documents generated by a variety of generators, including ChatGPT (cha, 2023), GPT-4 (OpenAI, 2023), Llama-2 (Touvron et al., 2023), Davinci(Brown et al., 2020), FlanT5 (Chung et al., 2022), Dolly (Conover et al., 2023), Dolly2 (Conover et al., 2023), BloomZ (Muennighoff et al., 2023), and Cohere[4]. Table 12 shows the number of texts for each domain.

F Further variations on main experiments

We repeat the single-target few-shot experiment reported in §4.4 where all episodes are comprised of only $N = 1$ document. The results are shown in Table 13.

In Table 16 and Table 17 we break down the results of the experiments reported in §4.4 and §4.5. respectively according evaluation dataset.

G Comparison with watermarking

Watermarking has emerged as a strong approach to admit the detection of text generated by a given language model by altering that model's token distributions according to a particular watermarking

[4] https://cohere.com

Data Type	Prompt Sources
AAC Data	r/politics and r/PoliticalDiscussion
LWD Data, Same Topic	r/politics and r/PoliticalDiscussion
LWD Data, Different Topic	r/anime, r/MMA, r/movies, r/personalfinance, r/soccer
LWD Data, Different Domains	Amazon Product Reviews, StackExchange Posts

Table 8: Data Descriptions.

LWD LMs	Same Topic	New Topic	Diff Domain
Machine Generated Texts	16,833	82,105	18,074
Human Written Texts	16,833	82,105	18,643
Total Texts	33,666	164,210	36,717

Table 9: LWD LM Data

Persona	Example Document
Human Prompt	And the politicians that needed relief in Sandy were opposed to relief during Katrina. And the politicians that needed relief during Katrina were opposed to relief for whatever disaster preceded it. Right now theres someone in, like, North Carolina reading about lawmakers who were opposed to Sandy relief but are now asking for relief in Louisiana, and theyre thinking Sure, this backfired on *them*, but I can safely vote against flood aid in Louisiana because theres no way this **exact** thing could happen to me!
Teenager	Ugh, it's just so frustrating how politicians only care about their own interests. It's like they forget that they're supposed to be serving the people and not just looking out for themselves. I wish they would stop playing politics with people's lives and actually do their jobs.
Concerned Citizen	As a concerned citizen, it's disheartening to see political leaders being hypocritical in times of need. We need to prioritize humanity over political affiliations and offer aid to those who suffer from natural disasters regardless of their location or political beliefs.
Skeptical Optimist	I understand why you might feel skeptical about politicians who were once opposed to disaster relief suddenly asking for it when their own state is affected. However, as an optimist, I believe it's important to focus on the fact that these politicians are now recognizing the need for assistance during times of crisis.
Pragmatic Realist	As a pragmatic realist, I understand that politics often involves inconsistent and hypocritical behavior from lawmakers. However, when it comes to disaster relief, it's important to separate politics from practicality. Regardless of a politician's stance on relief for previous disasters, the immediate needs of the current disaster should be addressed.
Passionate Activist	As an activist deeply committed to improving disaster relief policies, I find it appalling that politicians would maintain such hypocritical stances on disaster relief. We must recognize the need for and benefits of supporting our fellow Americans in times of crisis, regardless of their political affiliations or geographic location.

Table 10: Example documents dealing with politics, generated by ChatGPT and prompted according to the desired personality of the author in order to elicit diverse writing styles.

strategy (Kirchenbauer et al., 2023). Because this takes place when documents are generated, the approach requires direct access to the language model, a significant limitation relative to our proposed method, which has no such requirement. In this section we compare our approach to the statistical tests to detect watermarked text proposed in the work cited above, and further consider a simple remediation that adversaries may deploy to circumvent watermarking.

We adapt the watermarking procedure outlined by Kirchenbauer et al. (2023) to apply watermarks to text generated by Llama-2 (Touvron et al., 2023). For this, we follow the same prompting procedure discussed in Appendix E to generate documents in the Amazon domain, which we attempt to distinguish from real Amazon reviews using both our proposed approach and the statistical approach outlined in the reference above. The results shown in Figure 5 show that given a reasonable amount of text, our approach outperforms statistical tests to detect watermarked text.

An adversary may apply paraphrasing to watermarked documents in order to make them less easy to detect. To simulate this, we use DIPPER to paraphrase each watermarked document, setting the lexical diversity parameter to 20%. We repeat the experiment above using the paraphrased documents in place of the original watermarked documents. The results shown in Figure 5 suggest that our approach is considerably more robust to paraphrasing applied as remediation to watermarking.

Dataset	Human Example	ChatGPT Example
Amazon	The kids immediately wanted to put on a puppet show with this (they received a box full of puppets separately), but no doubt this will also be used for a store and other things.	As an above-average fixed location-fixed view security camera, it provides SD quality images that can be conveniently viewed from the screen of a smart phone.
ArXiv Abstract	We introduce a density tensor hierarchy for open system dynamics, that recovers information about fluctuations lost in passing to the reduced density matrix.	The motivation for this research comes from the challenges encountered in simulating flows with strong nonequilibrium effects, such as flows with high-speed micro-jets, turbulent mixing, and multiphase flows.
Reddit ELI5	For example, this is why there is a differentiation between being depressed (aka a depressive episode) and being diagnosed with major depressive disorder.	Now, as you may know, the Wright brothers - Orville and Wilbur - are widely credited with inventing the airplane.
Wikipedia	During the reign of the Shah kings, the Mulkajis (Chief Kajis) or Chautariyas served as prime ministers in a council of 4 Chautariyas, 4 Kajis, and sundry officers.	Born in Howard County, Maryland, on December 10, 1832, Carroll was the son of John Carroll, a prominent lawyer and politician from Maryland.
Wikihow	You will not always be the most intelligent person in the room, and the farther you get from school, the less book smarts will matter in your day-to-day life.	Whether you want to create a collage of memories for a special occasion or display your favorite photos in an artistic way, Inkscape can make it happen.

Table 11: Examples of human text and ChatGPT generations from the Amazon and M4 datasets, truncated to a maximum of 32 tokens.

M4	Peerread	ArXiv Abstract	Reddit ELI5	Wikihow	Wikipedia
Machine Generated Texts	13,831	17,340	15,885	14,901	13,677
Human Written Texts	5,203	2,997	2,999	2,999	2,975
Total Texts	19,034	20,337	18,884	17,900	16,652

Table 12: M4 domains and statistics.

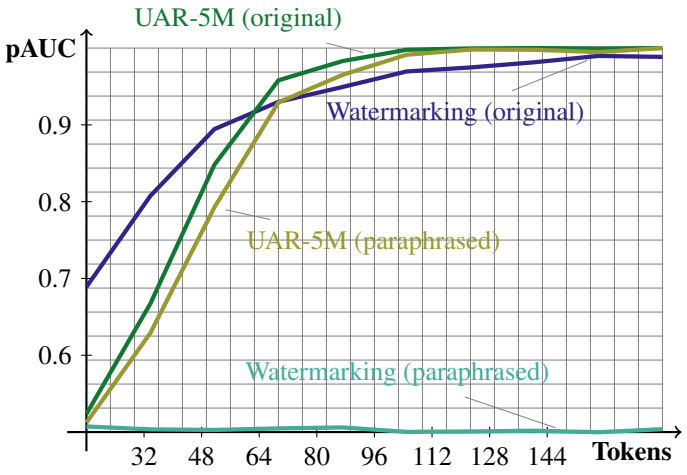


Figure 5: pAUC of the proposed approach and the watermark detector as the number of tokens is varied for the Amazon dataset. The proposed approach is more robust to paraphrase attacks, and achieves equal or better results to the watermark detector when the number of tokens is ≥ 48.

Method	Training	pAUC	
UAR	Reddit (5M)	**0.687 (0.001)**	**0.711 (0.002)**
UAR	Reddit (5M), Twitter, StackExchange	**0.687 (0.001)**	**0.710 (0.002)**
UAR	AAC, Reddit (politics)	0.651 (0.001)	0.657 (0.002)
CISR	Reddit (hard neg/hard pos)	0.607 (0.001)	0.613 (0.002)
RoBERTa (Zero shot)	AAC, Reddit (politics)	0.651 (0.031)	0.659 (0.033)
RoBERTa (ProtoNet)	AAC, Reddit (politics)	0.535 (0.001)	0.535 (0.001)
RoBERTa (MAML)	AAC, Reddit (politics)	0.526 (0.002)	0.526 (0.002)
SBERT	Reddit	0.552 (0.001)	0.563 (0.001)
Random		0.005	0.005

Table 13: Results of the single-target few-shot experiment reported in §4.4 where all episodes are comprised of $N = 1$ document.

H Effect of shorter truncation

We now repeat the experiment described in §4.4 where we truncate all support samples and queries to the nearest sentence boundary before the thirty-second token. This is in contrast with the truncation applied in the body of this paper, where we truncated all samples and queries to the nearest sentence boundary before the hundred-twenty-eight token. We also report two additional metrics, namely the usual area under the ROC curve (AUC) and the false positive rate corresponding with a 95% true positive rate (FPR@95TPR). These results are reported in Table 14.

Method	Training Dataset	AUC	pAUC	FPR@95
Few-Shot Methods				
LUAR	Reddit(5M)	**0.9884**	0.9094	**0.0533**
LUAR	Reddit (5M), Twitter, StackExchange	**0.9884**	**0.9118**	0.0543
LUAR	AAC, Reddit (politics)	0.8913	0.7308	0.3394
CISR	Reddit (hard neg/hard pos)	0.9608	0.7973	0.1440
RoBERTa (ProtoNet)	AAC, Reddit (politics)	0.9791	0.9062	0.0934
RoBERTa (MAML)	AAC, Reddit (politics)	0.6686	0.5141	0.6555
SBERT	Multiple	0.9673	0.8087	0.1448
Zero-Shot Methods				
RoBERTa (Zero shot)	AAC, Reddit (politics)	0.7238	0.5295	0.5369
OpenAI Detector	WebText, GPT2-XL	0.6933	0.5559	0.7698
Rank (GPT2-XL)	BookCorpus, WebText	0.7301	0.5423	0.6341
LogRank (GPT2-XL)	BookCorpus, WebText	0.9107	0.6395	0.2209
Entropy (GPT2-XL)	BookCorpus, WebText	0.2083	0.4977	0.9667
Random		0.5000	0.0500	

Table 14: Mean of AUC, pAUC and FPR@95, for various few-shot and zero-shot approaches trained using various combinations of domains, topics, and LLM. Each model was evaluated on a common corpus of documents consisting of unseen domains, topics, and LLM, organized into episodes of $N = 10$ documents.

I Detection of unknown LLM

Our final experiment follows the procedure described in §4.4 with the following modification. To create support samples, we randomly sample N documents from among the documents in the evaluation dataset generated by LLM. Thus, a support sample need not consist of documents by a single LLM, although the query episodes *do* consist of documents generated by a single author, either LLM or human. This experiment reflects the situation where a handful of documents are known to have originated from LLM, but the specific LLM generating each document cannot be attributed. As before, the detector score reflects the likelihood that a given query was generated by *any* LLM contributing to the evaluation corpus. The results are reported in Table 15.

Method	Training Dataset	pAUC $N = 1$	pAUC $N = 2$
UAR	Reddit (5M)	0.682 (0.002)	0.759 (0.002)
UAR	Reddit (5M), Twitter, StackExchange	0.684 (0.002)	0.706 (0.002)
UAR	AAC, Reddit (politics)	0.640 (0.002)	0.633 (0.002)
CISR	Reddit (hard neg/hard pos)	**0.707 (0.003)**	**0.779 (0.003)**
AI Detector (fine-tuned)	AAC, Reddit (politics)	0.660 (0.029)	0.668 (0.031)
RoBERTa (ProtoNet)	AAC, Reddit (politics)	0.536 (0.001)	0.524 (0.001)
RoBERTa (MAML)	AAC, Reddit (politics)	0.672 (0.007)	0.724 (0.010)
SBERT	Multiple	0.552 (0.001)	0.546 (0.001)
Random		0.005	0.005

Table 15: Results on detection of unknown LLM.

Dataset	N	UAR	UAR Multi-LLM	UAR Multi-domain	CISR	AI Detector (fine-tuned)	RoBERTa (MAML)	RoBERTa (ProtoNet)	SBERT	Random
Amazon	5	0.998	0.999	1.000	0.986	0.581	0.891	0.998	0.648	0.05
	10	1.000	1.000	1.000	1.000	0.582	0.886	1.000	0.841	0.05
Reddit	5	1.000	0.999	1.000	0.975	0.992	1.000	0.998	0.911	0.05
	10	1.000	1.000	1.000	0.989	0.865	0.943	0.999	0.972	0.05
M4 Arxiv	5	0.989	0.951	0.890	0.996	0.659	0.702	0.951	0.653	0.05
	10	1.000	0.990	0.975	1.000	0.641	0.731	0.995	0.713	0.05
M4 PeerRead	5	0.829	0.850	0.914	0.819	0.737	0.643	0.885	0.623	0.05
	10	0.946	0.919	0.977	0.946	0.788	0.683	0.965	0.738	0.05
M4 Reddit	5	0.819	0.900	0.935	0.723	0.613	0.855	0.841	0.608	0.05
	10	0.890	0.974	0.989	0.854	0.610	0.845	0.920	0.657	0.05
M4 WikiHow	5	0.871	0.828	0.862	0.659	0.499	0.538	0.799	0.659	0.05
	10	0.980	0.920	0.964	0.814	0.499	0.539	0.918	0.757	0.05
M4 Wikipedia	5	0.845	0.777	0.801	0.753	0.718	0.670	0.747	0.519	0.05
	10	0.979	0.878	0.930	0.919	0.715	0.700	0.866	0.566	0.05

Table 16: pAUC for single target detection experiment in §4.4, broken down by evaluation dataset.

We continue to see high detection accuracies in this detection setting, with the methods based on style representations outperforming other baselines. We now find that CISR performs better than other approaches, although the UAR variants remain competitive. Note that this experiment introduces a train-test mismatch for UAR, since each episode used to train UAR consists of documents by the same author, in contrast with the support samples used in the experiment, which typically contain documents by *multiple* LLMs.

Dataset	N	UAR	UAR Multi-LLM	UAR Multi-domain	CISR	RoBERTa (ProtoNet)	SBERT	Random
Amazon	5	0.999	0.999	1.000	0.999	0.995	0.597	0.05
	10	1.000	1.000	1.000	1.000	1.000	0.699	0.05
Reddit	5	1.000	0.997	1.000	0.937	0.959	0.906	0.05
	10	1.000	1.000	1.000	0.984	0.999	0.998	0.05
M4 Arxiv	5	0.981	0.992	0.982	0.961	0.503	0.774	0.05
	10	1.000	1.000	1.000	0.999	0.502	0.928	0.05
M4 PeerRead	5	0.665	0.994	0.733	0.500	0.500	0.559	0.05
	10	0.935	0.998	0.974	0.517	0.499	0.705	0.05
M4 Reddit	5	0.773	0.937	0.990	0.499	0.498	0.634	0.05
	10	0.897	1.000	1.000	0.499	0.497	0.664	0.05
M4 WikiHow	5	0.969	0.968	0.991	0.505	0.498	0.942	0.05
	10	0.999	0.995	0.998	0.522	0.497	0.990	0.05
M4 Wikipedia	5	0.907	0.956	0.794	0.521	0.502	0.498	0.05
	10	0.988	0.999	0.969	0.644	0.503	0.500	0.05

Table 17: pAUC for multi-target detection experiment in §4.5, broken down by evaluation dataset.