How well can machine-generated texts be identified and can language models be trained to avoid identification?

Sinclair Schneider Florian Steuber João A. G. Schneider Gabi Dreo Rodosek
Bundeswehr University Munich, Bavaria, Germany
{Sinclair.Schneider, Florian.Steuber, Joao.Schneider, Gabi.Dreo}@unibw.de

Abstract

With the rise of generative pre-trained transformer models such as GPT-3, GPT-NeoX, or OPT, distinguishing human-generated texts from machine-generated ones has become important. We refined five separate language models to generate synthetic tweets, uncovering that shallow learning classification algorithms, like Naive Bayes, achieve detection accuracy between 0.6 and 0.8.

Shallow learning classifiers differ from human-based detection, especially when using higher temperature values during text generation, resulting in a lower detection rate. Humans prioritize linguistic acceptability, which tends to be higher at lower temperature values. In contrast, transformer-based classifiers have an accuracy of 0.9 and above. We found that using a reinforcement learning approach to refine our generative models can successfully evade BERT-based classifiers with a detection accuracy of 0.15 or less.

Keywords: Language Models, Language Model Detection, Transformer Reinforcement Learning

1. Introduction

Improving transformer models has led to the creation of higher-quality machine-generated text. This has led to the question of whether a distinction between human-written and machine-generated text is reliably possible. The main concern is that it can be difficult for humans to make an accurate distinction because they focus on linguistic properties of the text rather than statistical features such as word probability, which are the focus of classification models.

According to Ippolito et al. (2020), humans are better in their judgment if the number of unlikely words increases, whereas classification models exhibit the opposite behavior, prioritizing statistical evidence. Similar studies, including Gehrmann et al. (2019), concluded that humans can detect fake texts with an accuracy of 54%. Without computer-aided tools, humans achieve only a 50% accuracy rate in identifying GPT-3 generated texts (Clark et al., 2021).

These findings highlight the need for a machine-guided decision process in reliably identifying artificially generated texts. This work, therefore, aims to investigate the accuracy and reliability of detection mechanisms when applied to GPT (Generative Pre-trained Transformer)-generated texts.

We conducted a study that involved comparing different temperatures, sampling methods, and sample sizes using basic classification algorithms. The outcome revealed that these methods were insufficient in distinguishing between human-written and machine-generated texts. However, when more advanced classifier models like BERT were utilized, the results were more consistent and reliable.

Various approaches exist to trick or bypass these detection mechanisms, including paraphrasing the generated texts (Krishna et al., 2023). Introducing an alternative approach, we focus on transformer-based reinforcement learning to bypass the detection classifier. This technique was initially proposed by Ziegler et al. (2020) to fine-tune language models using human feedback. In contrast to the conventional training approach, we treat the feedback of the detector model as a reward during the reinforcement learning processing. Consequently, the generator model is rewarded most when the classifier incorrectly identifies the output as human-generated. We furthermore add additional linguistic constraints to refine the generated texts,

URI: https://hdl.handle.net/10125/106711
978-0-9981331-7-1

avoiding exploitation of the classification model. This points out the limitations of detector models in detecting intentionally altered texts created by generative models that have been modified to evade the classifier by malicious actors.

2. Related Work

2.1. Automatic Generation of Texts

Different approaches have been used to create language models capable of generating texts. The most common architecture revolves around transformer models, including GPT and its predecessors, such as GPT-Neo-125M, GPT-Neo-1.3B, GPT-Neo-2.7B (Black et al., 2021), GPT-J-6B (B. Wang & Komatsuzaki, 2021), OPT-125M, OPT-350M, OPT-1.3B, OPT-2.7B (Zhang et al., 2022) and GPT-2 (Radford et al., 2019). Further variants include Instruct-GPT (Ouyang et al., 2022) and T5 from Google (Raffel et al., 2020) model.

2.2. Detection Techniques

Orabi et al. (2020) provide a further taxonomy of detection techniques, based on *graphs* (e.g. Daya et al., 2020), *crowdsourcing* (i.e. manually bot identification, e.g. G. Wang et al., 2013), *anomalies* (e.g. Nomm & Bahsi, 2018) and *machine learning* (Alothali et al., 2018).

However, since most of the approaches are either carried out by humans or make use of automatic tools, one can simplify them into two categories, i.e., *human* vs. *machine-based techniques*[1]. Both of them are based on *behavior* vs. *content* of the user. We focus on pure content, which is strictly speaking *text* (in the case of Twitter posts, so-called *tweets*). (Alothali et al., 2018)

Human-based Detection. When tasked to identify computer-generated documents, human individuals mainly focus on the semantic coherence of the presented texts. This contrasts machine-based detection approaches, which instead focus on statistical properties, such as word probabilities and sampling schemes (Ippolito et al., 2020). Dugan et al. (2020) present a tool to evaluate human detection and conclude that it is relatively easy to fool humans for the above reasons.

Machine-based Detection. Statistical methods for machine-generated text detection are based on the fact that different modeling techniques leave detectable artifacts in the generated texts (Tay et al., 2020).

Furthermore, there exist various rule-based models to detect automatically generated texts, which are based on improbable word sequences and grammar (Cabanac & Labbé, 2021) or on similarity measures including word overlap (Harada et al., 2021).

RoBERTa or BERT-based classifiers have downsides, such as their tendency to over-fit and the need to train a classification model every time a new generator is released. To solve this issue, zero-shot classifiers like DetectGPT (Mitchell et al., 2023) have been developed, requiring only a duplicate of the model that should be tested and a second language model to introduce random permutations to the test-string. Due to their principle of introducing random permutations to the test string, they cannot operate with a short input string.

To conclude, due to the novelty of most classification approaches, there are few scientific publications on bypassing them, like the paraphrasing-based one from Krishna et al. (2023). Since we use reinforcement learning to adjust our generative model to bypass the classifier, our approach is based on the paper "Fine-Tuning Language Models from Human Preferences" by Ziegler et al. (2020).

3. Methodology

Our methodology is structured as follows. We describe the data set used for and the training of generator models. Then, we investigate different approaches for machine-generated text detection, starting with shallow learning methods and ranging to transformer-based detectors. We present our reinforcement learning model aimed at generating tweets that evade previous detection approaches.

3.1. Data set

Our data set consists of tweets recorded between January and February 2020. The data is saturated with spam and advertising content, making it poorly suitable as direct training input. We, therefore, applied a set of filter policies.

First, we filter tweets composed in English to achieve comparable results when evaluating against other state-of-the-art literature. Next, we extracted texts of verified users with less than $\geq$ 100.000 followers to avoid accounts that promote advertisements. Subsequently, we choose non-truncated tweets, as longer text might get truncated if not fully requested, which is undesirable for training. Furthermore, we restrict to original tweets and omit quotes, replies, and

[1] this is true at least for all the mentioned exemplary studies, even though Orabi et al. (2020) use different terminology, e.g. graph-based, anomaly-based, etc.

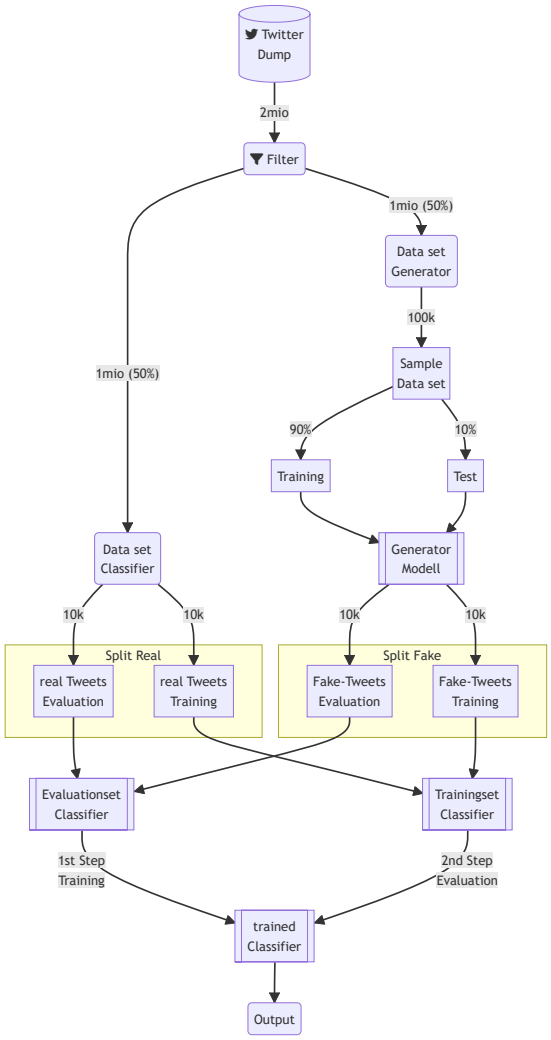

Figure 1. Data pipeline used for modeling

retweets to obtain a data set of more diverse texts. Finally, we check if the author of the tweets exhibits an average tweet volume (at most 20 tweets per day) to avoid spam content.

Overall, the data set consists of $k = 2.000.000$ tweets from $n = 136.450$ real accounts and is split into two equal parts, one for the transfer learning of the language models and one for testing the classifiers. We complement both training and evaluation data sets by adding real-world tweets. The generated tweets then form the two data sets "Fake-Tweets Evaluation" and "Fake-Tweets Training". The counterparts ("Real Tweets Evaluation" and "Real Tweets Training") are then built out of the original filtered tweets as illustrated in Figure 1.

3.2. Training of Generative Models

Our initial focus is to examine the efficacy of a fine-tuned LLM when generating tweets, as we believe they are more challenging to create than texts containing longer passages. Additionally, we aim to assess the capability of various existing detectors in identifying machine-generated short texts. Our experiments range from simple methods such as Naive Bayes to more advanced models such as BERT.

As a basis for our research, we make use of various GPT variants, including GPT-2 (Radford et al., 2019), GPT-J-6B (B. Wang & Komatsuzaki, 2021), GPT-Neo-125M, GPT-Neo-1.3B, GPT-Neo-2.7B (Black et al., 2021), OPT-125M, OPT-350M, OPT-1.3B and OPT-2.7B (Zhang et al., 2022). Despite the availability of larger free models, we opt to utilize transfer learning-compatible models on a machine equipped with an A6000 GPU and 128GB of RAM.

Each fine-tuned model generates two sets of fake tweets with a given sampling strategy and temperature. These are used to train and evaluate the detector model. The size of the generated sets varies, ranging from 10.000 for comparisons of simple Bag-of-Words classifiers to 100.000 for training BERT classifiers. Examples of generated tweets can be taken from Table 1.

3.3. Shallow Detectors for machine-generated Texts

We begin by exploring the effectiveness of a Naive Bayes classifier using Bag-of-Words (BoW) feature inputs to identify synthetic tweets. This classifier is chosen due to its simplicity and short training time. Due to the restricted number of features that BoW extracts from the texts, modifications in the generation parameters of language models, such as sampling methods, k-values, p-values, etc., exert a significantly greater influence on the classifier's output.

We aim to examine how the parameters of the generator, namely temperature, sampling strategy, sampling size, and model size, influence the accuracy of the classifier. For this, we performed a grid-based approach in training the classifier with different parameterizations, each time conducted using 10,000 real and generated tweets.

Temperature. A generator's temperature τ alters the probability distribution of the next word. A low value sharpens the distribution curve, resulting in more accurate sentences. In contrast, a higher value flattens the curve and thus allows a higher variety of words to

Table 1. Example tweets generated with different models with temperature = 1.0 and top-50 sampling

Model	Tweet
GPT-Neo-125M	Kobe says new coronavirus warning on plane is too difficult to understand.
OPT-125M	I'm sure a few will be added in a future update as part of the "Duke" legacy.
OPT-350M	Good luck on the final stage of your tour!
GPT-Neo-1.3B	The new album is out now; make sure you have the album download code for free.
OPT-1.3B	Rangers' Henrik Lundqvist: "I'm not even thinking about' the trade rumors, says the goalie"
GPT-Neo-2.7B	#ValentinesDay: Today is the day to celebrate the greatness of yourself. And to appreciate...
OPT-2.7B	A very cold, chilly #day for #Lincoln and #Omaha #MorningWeather
GPT-J-6B	"This is how we play games!" Let's hear "The Box" tonight with @OzzyOsbourne...

be chosen, increasing the outcome's variance at the cost of linguistic coherence. In our experiments, we vary temperature values ranging from 0.8 to 1.4.

Sampling strategy. LLMs produce sentences by generating one token at a time. Token selection depends on the selected sampling method. For example, greedy search selects the token with the highest probability as the predecessor, whereas random sampling strategies pick the next token based on its probability. This process can be limited to a certain number (k-sampling) or a combined probability (p-sampling) of the most probable next tokens. Furthermore, to enhance the unpredictability of word generation, one can employ the typical sampling technique using conditional entropy (Meister et al., 2023).

Sampling size. It is well known that both shallow and deep machine learning models yield better results when trained on larger data sets. This improvement diminishes with a certain amount of data depending on the complexity of the model. We test each classification algorithm with a training set of varying sizes (1k, 10k, 50k, and 100k tokens) to investigate the correlation and saturation.

Model size. OpenAI's GPT family has achieved notable language modeling advancements through substantial parameter size increases. While larger amounts are expected to generate better text results, they are more effective in creating longer texts. This is due to the larger context window associated with more complex models. Since our experiment involves generating short tweets, we aim to investigate if the model size affects the detectability of the language model. Our study compares models ranging from 125 million to 6 billion parameters.

3.4. Transformer-based Detectors for machine-generated Texts

Given their state-of-the-art capabilities in text classification, transformer-based detectors are more likely to be used in production. We opted to use BERT as the primary reference model in the transformer family. Its performance is nearly on par with more advanced versions such as RoBERTa, as assessed by the MultiNLI benchmark (Williams et al., 2018). In their evaluation, BERT-Large records a score of 88% (Lee-Thorp et al., 2022), RoBERTa at 90.8% (Liu et al., 2019), and DeBERTa at 91.1% (He et al., 2021).

The evaluation process is similar to the shallow detectors, except that BERT's embeddings replace the BoW representation. Because of the training complexity, we refrain from fitting multiple BERT models with varying parameters.

3.5. Reinforcement Learning to bypass the Detector

Our final contribution shows that even an advanced detection model such as BERT can be bypassed. We utilize a reinforcement learning approach for text generation that progressively learns from the detector output of previously generated texts. This approach roughly consists of three steps: rollout, evaluation, and optimization (von Werra et al., 2020), as discussed below.

The first step includes the model's rollout. Here, the language models from Section 3.2 generate an artificial tweet. For this, a model is provided with the initial part of an original tweet and is required to finish the sentence. In addition to completing tweets, it sometimes has to generate full texts independently to prevent overfitting on short texts.

Secondly, the evaluation step takes place. Texts and responses are supplied to the corresponding BERT classifier, as described in Section 3.4. Should a classifier identify the text as composed by humans, a positive reward is assigned to the reinforcement learning

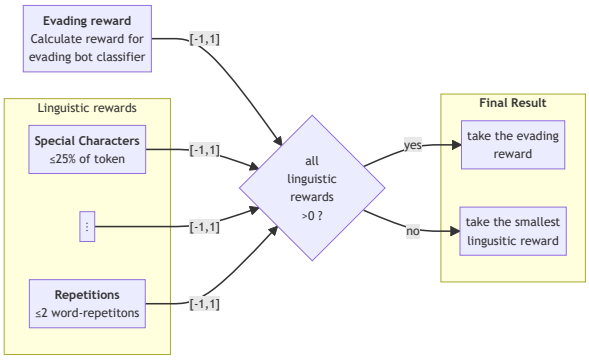

Figure 2. RL reward calculation

algorithm, otherwise a negative one. Raw logits tend to perform best in this case.

Finally, the optimization step takes place. The log probabilities of the tokens are calculated to compare the active language model to the reference model. This is part of the reinforcement learning described by Ziegler et al. (2020) and prevents the adjusted model from overoptimizing.

We now delve into more detail about our proposed handcrafted reward function. This reward supplements the detector's response and imposes a separate penalty on generated text, even if classified as human-written, in cases where specific linguistic constraints are not satisfied. Calculating rewards is depicted in Figure 2. If one or more of the supplementary rules listed below produce unfavorable results, the smallest one is chosen. Conversely, if all the additional rules yield favorable results, the value indicating that humans created the text is selected.

Special characters. A text should not include more than 25% of special characters. If it does, the model assigns a linearly decreasing negative reward of up to -1 if the text consists exclusively of special characters.

Repetitions. If a text contains more than two repetitions of the same word, it is assigned a negative reward of up to -1 when repeated eight times.

Linguistic acceptability. The linguistic acceptability is checked by the usage of a DeBERTaV3-v3-large classifier (He et al., 2023) trained on the Corpus of Linguistic Acceptability (CoLA; Warstadt et al., 2019). Since this CoLA data set contains sentences marked as acceptable or unacceptable by humans from a grammatical point of view, the trained DeBERTa model is able to judge how good or acceptable a given sentence is. If the value is below a threshold of 40%, a negative reward is returned, which minimizes to -1 at 0% acceptability. Note that we relax the threshold value to 30% when applied to models with over 2 billion parameters, as they are harder to train.

The thresholds mentioned are not taken over from a reference paper but are determined empirically. Generally, it is recommended to aim for a higher threshold as it directly affects the text's linguistic acceptability. Nevertheless, we discovered that setting the threshold too restrictively can lead to the failure of the reinforcement learning process. In such cases, the generated text might be classified as non-human, rendering it impossible for the model to obtain rewards and impeding its learning progress.

Dictionary. To obtain natural texts, at least 25% of tokens have to appear in a dictionary. Otherwise, a linearly increasing negative reward is assigned, which rises to -1 if none of the words appear in a dictionary. We remark that this value should be chosen relatively low in the context of tweets, as short texts typically contain more abbreviations and slang language.

Word Emoji relationship. A tweet must not contain more emojis than words. If it does, the reward is progressively penalized, up to a maximum, if only 25% of the tweet is formed by words. Emojis are not counted as special characters.

Number of Emojis. There should not be more than three emojis in one tweet. Every additional Emoji leads to a negative reward of -0.4 up to -1.

Repetition of the Query. To generate unique texts different from the input query, we assign a negative reward if more than half of the query is included in the output. This reward decreases to -1 if the entire query is repeated.

Special Token. There should not be more than two special tokens in one generated tweet, including BOS (beginning of the sentence) or EOS (end of sentence) delimiters. Every additional token gets a negative reward of 0.4 up to -1.

Same start. Sometimes, we require the model to generate tweets without an input query. In such cases, it is essential that the model still generates diverse outputs. Consequently, a negative reward is returned if more than 10% of the tweets within a single training batch start with the same word. This value again decreases linearly to -1 if 20% of all sentences have similar starts.

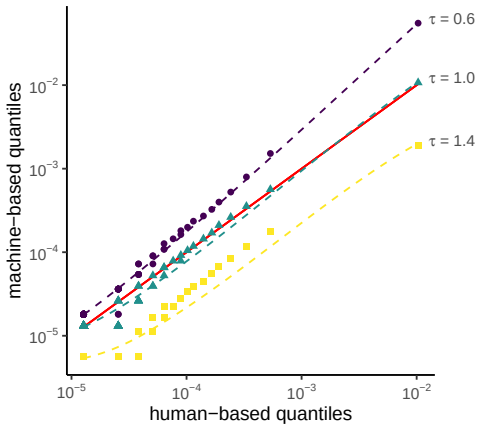

Figure 3. Humans' against machines' word probability distributions

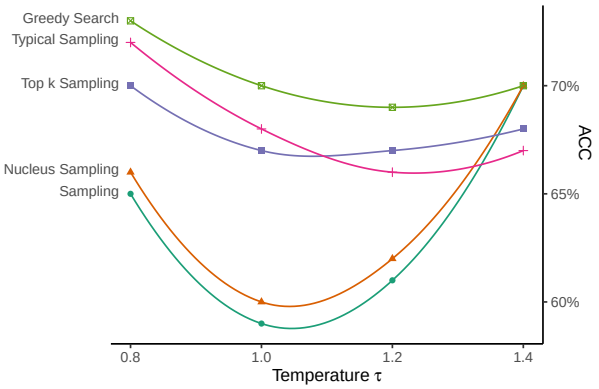

Figure 4. Sampling methods' detection rates by temperature (top k = 100, nucleus p = 0.95)

Numbers at the start. Like the previous policy, we do not want tweets to start with numbers frequently. Therefore, if the model grasps that a classifier can be circumvented by a sentence beginning with a number once, it may also learn that a similar sentence, or one with the same pattern, could be effective on another occasion. Hence, only 10% of the sentences within a training batch are allowed to start with a number, resulting in a negative reward of up to -1 if there are more.

Unknown characters. Occasionally, language models may insert filler characters, represented as unknown characters. This phenomenon typically occurs when unknown characters are included in the data set used for fine-tuning. To prevent the model from using these characters to bypass the classifier, generating them will result in a negative reward starting with -0.5 and decreasing if the replacement character appears more than once.

These additional optimization rules have been found by analyzing various preliminary runs and observing the logs for requests and responses during RL training. It is worth noting that RL can function without these rules, but the results are considerably inferior, as demonstrated by outcomes such as: "Something for Administrator930 Macy's Displays! RIP Family Members". In this particular case, the rule for linguistic acceptability would have avoided a positive reward.

For bigger models, the reward of the bot classifier can be multiplied by a factor, e.g., 10, to give the task of bypassing the classifier a higher priority than having excellent sentences.

3.6. Applicability to other text-domains

We conducted a second iteration using a CNN and Daily Mail data set (Hermann et al., 2015) to demonstrate the practical application of the reinforcement learning approach. This also proved that our approach can be adjusted to generate fake news. The training and testing procedure for this iteration was similar to the fake tweets, except the linguistic refinement filters were reduced to only one. This filter ensured that not all generated texts start the same.

4. Results

4.1. Evaluation of Shallow Detectors

Word Distributions. By altering the probability distributions of the generating language model, statistical-based classifiers like Naive Bayes can detect changes more easily. In Figure 3, a quantile-quantile plot visualizes the statistical differences that these classifiers rely on. The plot shows the probability distributions of words present in a corpus with different temperature values, which differ significantly from the word distributions in human-written texts.

This shows how machine-based detection differs significantly from human-based detection. A classifier can detect changes in the probability distribution in both directions, while humans rely on language comprehension. The level of linguistic acceptability increases when the distribution curve is sharpened and the probability of the most likely next words increases. However, reducing the temperature to achieve this comes at the expense of reducing the information content.

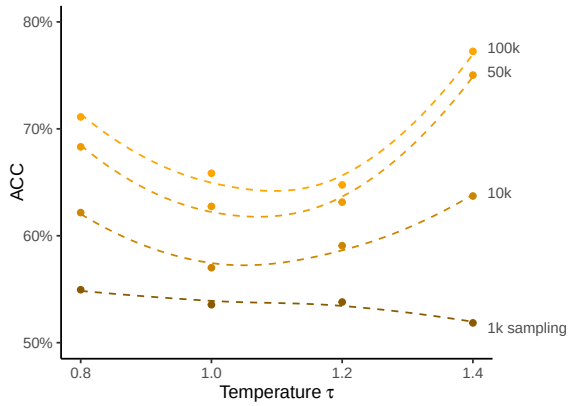

Figure 5. Comparison of different sampling sizes

Sampling Schemes. Next, we compare how different sampling methods used for generating texts affect the detection rates of shallow models. The results can be seen in Figure 4. The easiest method to detect is greedy search, selecting the most likely next word based on maximum likelihood. Typical-p sampling ranks second and relies on conditional entropy to deviate from the original distribution. Top-k sampling also differs significantly from the initial distribution as it does not consider the distribution curvature. Finally, nucleus sampling, which considers the various shapes of the probability curves for the next token, is most evasive after sampling using pure randomness.

Sampling Size. Shallow learning models, including Naive Bayes, require small amounts of training data compared to deep learning models such as BERT. In Figure 5, we have listed how different training data sizes affect a shallow detector's accuracy in successfully identifying machine-generated texts. As can be seen, even with a training batch consisting of as little as 1000 data points, the detector can still capture some dependencies. Increasing the size of the training data improves the detector's quality. However, the improvements successively diminish above a certain threshold, from 50k to 100k.

Contrary to the sample size, the parameter size of the evaluated models has a minor impact on the classification process, as seen in Figure 6. In this figure, we have grouped GPT and OPT model families in similar colors, where darker lines represent models with more internal parameters. One possible explanation for this behavior is that even though bigger models typically perform better on larger text passages due to larger context windows, the generation of short texts hardly benefits from this property. In addition, given that the generator family (OPT vs. GPT) leads to more significant differences in classification accuracy, we hypothesize that model design and training corpus inherently play a more crucial role in generating short texts than sheer model size.

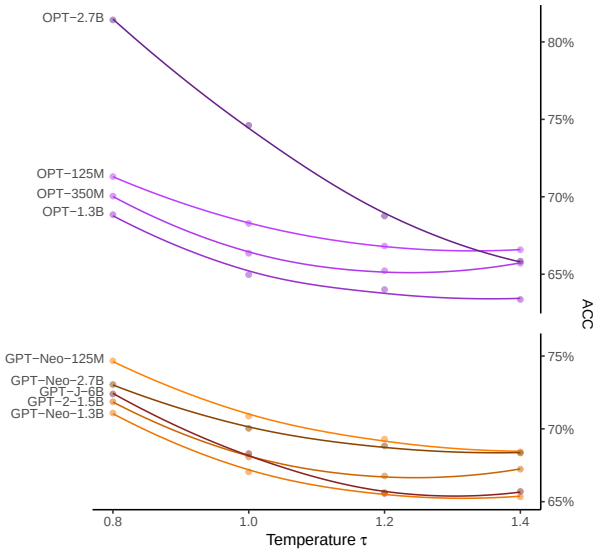

Figure 6. Generator models' ACC by temperatures

Table 2. F1-Scores before and after RL

		BERT F1-Scores	
Model	**parameter**	***pre***	***post***
GPT-Neo	125M	0.952	0.031
GPT-Neo	1.3B	0.933	0.146
GPT2	1.5B	0.929	0.090
GPT-Neo	2.7B	0.947	0.004
GPT-Neo*	2.7B	0.959	0.173

*Fake news based on CNN and Daily Mail data set and model

4.2. Evaluation of Transformer-based Detectors

The introduction of self-attention layers in transformer-based models has led to significant improvement in their performance. These layers allow the classifier to translate a token into vector space based on the surrounding tokens. As a result, all evaluated BERT models have achieved an F1-Score of over 0.9 (cf. Tab. 2, column *pre*). The parameters for tweet generation included top 50 sampling with a temperature of $\tau = 1.0$. While these results are impressive, there is a possibility that the BERT classifier might overfit one specific method and perform poorly on other generative models.

Table 3. Example of the reinforcement learning process

query	response	reward
FINAL	UPDATE: Chancellor says he has not shown anxiety	0.8019
<|startoftext|>	see you this weekend! we'll be here again cider is back in v	0.9674
<|startoftext|>	flush those lemons. This food ain't good	0.9628
Ozzy Osbourne cancel	s gig at studio theatre tonight over coronavirus 2020 forced	−0.0089
Today we'll know	soon morning if there's blood somewhere in	−0.1756
The best #adventure this	year on 2020, dudes. Can we really	−0.2899
<|startoftext|>	Yeah it's weird how local commentators	0.9696

4.3. Evaluation of Reinforcement Learning

Transformer-based classifiers are generally very reliable in distinguishing real and generated content, as investigated in the previous sections. However, we show that a carefully fine-tuned reinforcement learning approach can bypass these well-performing models. To obtain reliable results, much emphasis has to be given to optimizing the model's hyperparameters, including learning rate, mini-batch size, chosen optimizer, and threshold for detecting linguistic acceptability. Based on empirical analysis, experiments have shown that a learning rate of $5 \cdot 10^{-5}$ and a mini-batch size of 4 tend to yield the best results.

For the GPT-Neo-2.7B model, we reduced the linguistic acceptability threshold from 0.4 to 0.3 since bigger models are more susceptible to manual interventions. Similarly, we exchanged the Adam optimizer within the 2.7 billion parameter model with the novel Lion optimizer (Chen et al., 2023), reportedly outperforming the first one.

We shared the same sampling technique during the reinforcement learning and evaluation phases to ensure reliable results. Table 3 presents a snippet of the log that documents queries, responses, and rewards during the reinforcement learning process. As stated earlier, a positive reward is only possible if both the query and response have been classified as human and none of the supporting rules outlined in section 3.5 have yielded negative results. In this case, we solely rely on the BERT classifier's reward to detect generated text.

We list the classifier's F1-scores before and after the reinforcement learning process in Table 2. The 2.7 billion model had a lowered threshold for linguistic acceptability, resulting in significantly better results, but at the expense of lower linguistic quality. The experiment adapted to train the generator using CNN and the Daily Mail data set demonstrates the applicability of our approach to other text domains. Our experiment with four open-source models showed that a BERT classifier can be bypassed using a reinforcement learning training method.

5. Conclusions and Outlook

In this paper, we highlighted the significance of sampling techniques and the essential role of employing advanced models like transformers to detect generated texts accurately. We also demonstrated how a malicious actor could adapt generative models to evade a detector if accessible. However, this process is not simple and requires substantial computational power and hyperparameter tuning. As the parameter count of the generative model increases, the reinforcement learning process becomes more complex, and the output quality may decrease. While the risk of malicious actors modifying language models remains more theoretical than practical, it retains a degree of plausibility.

In various scenarios, such as completing homework in schools, submitting assignments at universities, or identifying suspicious campaigns on social media platforms like Twitter, it is essential to differentiate between human-written and machine-generated texts. OpenAI previously provided a detection model for this purpose, but it was discontinued in July 2023 due to inconsistency in distinguishing between the two (Kirchner et al., 2023). Our experiments have revealed a trade-off between creating easily identifiable texts that lack diversity and generating more diverse texts that are harder to differentiate but might not be as readable to humans. Therefore, larger AI-generated models may be more challenging to spot as they can combine both linguistic acceptability and diversity, thanks to their size and thus the ability to generate a higher range of plausible following tokens while still producing high-quality texts. Consequently, the trade-off between high-quality texts and low detectability is expected to diminish.

Therefore, it is essential to consider that language models may become undetectable. In such a scenario, students using ChatGPT to complete their homework could be viewed as a lesser evil. This could lead to the creation of customized social engineering bots that steal personal data without human input from the attacker, the development of malicious code by those without coding

knowledge, and customized spear phishing emails.

Unfortunately, when created using a language model, phishing emails may not display the typical red flags, such as poor grammar or spelling errors. This means that an attacker from anywhere in the world can create a convincing email in the language of the victim without paying for a translator.

Based on promising approaches such as DetectGPT (Mitchell et al., 2023) but also the withdrawal of the OpenAI classification model and our findings, we would like to emphasize that detecting and preventing these types of incidents is an emerging research field with ample opportunities for further publications.

Acknowledgements

The authors would like to thank the Chair for Communication Systems and Network Security for their valuable discussions and feedback as well as the Research Institute CODE for providing hardware.

References

Alothali, E., Zaki, N., Mohamed, E. A., & Alashwal, H. (2018). Detecting social bots on twitter: A literature review. *International Conference on Innovations in Information Technology (IIT)*, 175–180. https://doi.org/10/ggwcmw

Black, S., Leo, G., Wang, P., Leahy, C., & Biderman, S. (2021). *GPT-Neo: Large scale autoregressive language modeling with mesh-tensorflow* (Version 1.0) [Large language model]. Zenodo. https://doi.org/10/kqrc

Cabanac, G., & Labbé, C. (2021). Prevalence of nonsensical algorithmically generated papers in the scientific literature. *Journal of the Association for Information Science and Technology*, *72*(12), 1461–1476. https://doi.org/10/gj7b8h

Chen, X., Liang, C., Huang, D., Real, E., Wang, K., Liu, Y., Pham, H., Dong, X., Luong, T., Hsieh, C., Lu, Y., & Le, Q. V. (2023). *Symbolic discovery of optimization algorithms*. arXiv: abs/2302.06675. https://doi.org/10/kqzw

Clark, E., August, T., Serrano, S., Haduong, N., Gururangan, S., & Smith, N. A. (2021). All that's 'human' is not gold: Evaluating human evaluation of generated text. *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL/IJCNLP)*, 7282–7296. https://doi.org/10/kq2w

Daya, A. A., Salahuddin, M. A., Limam, N., & Boutaba, R. (2020). BotChase: Graph-based bot detection using machine learning. *IEEE Transactions on Network and Service Management*, *17*(1), 15–29. https://doi.org/10/gndzsg

Dugan, L., Ippolito, D., Kirubarajan, A., & Callison-Burch, C. (2020). RoFT: A tool for evaluating human detection of machine-generated text. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 189–196. https://doi.org/10/knkm

Gehrmann, S., Strobelt, H., & Rush, A. (2019). GLTR: statistical detection and visualization of generated text. In M. R. Costa-jussà & E. Alfonseca (Eds.), *Proceedings of the 57th annual meeting of the association for computational linguistics* (pp. 111–116). Association for Computational Linguistics. https://doi.org/10/gg9p2b

Harada, A., Bollegala, D., & Chandrasiri, N. P. (2021). Discrimination of human-written and human and machine written sentences using text consistency. *International Conference on Computing, Communication, and Intelligent Systems*, 41–47. https://doi.org/10/kq2z

He, P., Gao, J., & Chen, W. (2023). *DeBERTaV3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing*. arXiv: abs/2111.09543. https://doi.org/10/krs4

He, P., Liu, X., Gao, J., & Chen, W. (2021). *DeBERTa: Decoding-enhanced BERT with disentangled attention*. arXiv: abs/2006.03654. https://doi.org/10/krs5

Hermann, K. M., Kočiský, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., & Blunsom, P. (2015). Teaching machines to read and comprehend. In C. Cortes, D. Lee, M. Sugiyama, & R. Garnett (Eds.), *Proceedings of the 28th international conference on neural information processing systems* (pp. 1693–1701). MIT Press.

Ippolito, D., Duckworth, D., Callison-Burch, C., & Eck, D. (2020). Automatic detection of generated text is easiest when humans are fooled. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 1808–1822. https://doi.org/10/knkk

Kirchner, J. H., Ahmad, L., Aaronson, S., & Leike, J. (2023, January 31). *New AI classifier for indicating AI-written text* [Product Announcement]. OpenAI. Retrieved August

31, 2023, from https://openai.com/blog/new-ai-classifier-for-indicating-ai-written-text

Krishna, K., Song, Y., Karpinska, M., Wieting, J., & Iyyer, M. (2023). *Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense*. arXiv: abs/2303.13408. https://doi.org/10/kqzx

Lee-Thorp, J., Ainslie, J., Eckstein, I., & Ontañón, S. (2022). FNet: Mixing tokens with fourier transforms. *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 4296–4313. https://doi.org/10/grsfsm

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). *RoBERTa: A robustly optimized BERT pretraining approach*. arXiv: abs/1907.11692. https://doi.org/10/gp5knh

Meister, C., Pimentel, T., Wiher, G., & Cotterell, R. (2023). Locally typical sampling. *Transactions of the Association for Computational Linguistics*, *11*, 102–121. https://doi.org/10/kqrt

Mitchell, E., Lee, Y., Khazatsky, A., Manning, C. D., & Finn, C. (2023). *DetectGPT: Zero-shot machine-generated text detection using probability curvature*. arXiv: abs/2301.11305. https://doi.org/10/grsgh4

Nomm, S., & Bahsi, H. (2018). Unsupervised anomaly based botnet detection in IoT networks. *17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 1048–1053. https://doi.org/10/kq34

Orabi, M., Mouheb, D., Al Aghbari, Z., & Kamel, I. (2020). Detection of bots in social media: A systematic review. *Information Processing & Management*, *57*(4), Article 102250. https://doi.org/10/ghfkw8

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P. F., Leike, J., & Lowe, R. (2022). Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, & A. Oh (Eds.), *Advances in neural information processing systems* (pp. 27730–27744). Curran Associates.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). *Language models are unsupervised multitask learners* [OpenAI Blog].

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., & Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, *21*(1), 5485–5551.

Tay, Y., Bahri, D., Zheng, C., Brunk, C., Metzler, D., & Tomkins, A. (2020). Reverse engineering configurations of neural text generation models. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 275–279. https://doi.org/10/kq24

von Werra, L., Belkada, Y., Tunstall, L., Beeching, E., Thrush, T., & Lambert, N. (2020). *TRL: Transformer reinforcement learning* (Version v0.6.0) [Python package]. GitHub. https://github.com/lvwerra/trl

Wang, B., & Komatsuzaki, A. (2021). *GPT-J-6B: A 6 billion parameter autoregressive language model* [Large language model]. GitHub. https://github.com/kingoflolz/mesh-transformer-jax

Wang, G., Mohanlal, M., Wilson, C., Wang, X., Metzger, M. J., Zheng, H., & Zhao, B. Y. (Eds.). (2013). Social turing tests: Crowdsourcing sybil detection. In *Proceedings of the 20th annual network & distributed system security symposium*. https://doi.org/10/kq3v

Warstadt, A., Singh, A., & Bowman, S. R. (2019). Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, *7*, 625–641. https://doi.org/10/ggv8kd

Williams, A., Nangia, N., & Bowman, S. R. (2018). A broad-coverage challenge corpus for sentence understanding through inference. *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1112–1122. https://doi.org/10/gg9fnp

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M. T., Li, X., Lin, X. V., Mihaylov, T., Ott, M., Shleifer, S., Shuster, K., Simig, D., Koura, P. S., Sridhar, A., Wang, T., & Zettlemoyer, L. (2022). *OPT: open pre-trained transformer language models*. arXiv: abs/2205.01068. https://doi.org/10/kfxh

Ziegler, D. M., Stiennon, N., Wu, J., Brown, T. B., Radford, A., Amodei, D., Christiano, P., & Irving, G. (2020). Fine-tuning language models from human preferences. https://doi.org/10/gskffn