Data & Policy (2025), 7: e29
doi:10.1017/dap.2025.6

RESEARCH ARTICLE

Detection avoidance techniques for large language models

Sinclair Schneider, Florian Steuber, João A.G. Schneider and Gabi Dreo Rodosek

Bundeswehr University Munich, Munich, Bavaria, Germany
Corresponding author: Sinclair Schneider; Email: Sinclair.Schneider@unibw.de

Received: 30 September 2024; **Revised:** 11 January 2025; **Accepted:** 21 January 2025

Keywords: language models; language model detection; transformer reinforcement learning; paraphrasing attack; detection avoidance techniques

Abstract

The increasing popularity of large language models has not only led to widespread use but has also brought various risks, including the potential for systematically spreading fake news. Consequently, the development of classification systems such as DetectGPT has become vital. These detectors are vulnerable to evasion techniques, as demonstrated in an experimental series: Systematic changes of the generative models' *temperature* proofed *shallow learning—detectors* to be the least reliable (Experiment 1). Fine-tuning the generative model via *reinforcement learning* circumvented *BERT-based—detectors* (Experiment 2). Finally, *rephrasing* led to a >90% evasion of *zero-shot—detectors* like DetectGPT, although texts stayed highly similar to the original (Experiment 3). A comparison with existing work highlights the better performance of the presented methods. Possible implications for society and further research are discussed.

Policy Significance Statement

Large language models produce texts that appear indistinguishable from human ones, which is why research focuses on machine learning-based detectors. This article demonstrates how various state-of-the-art detectors can be tricked using different techniques. Text-generating models are specifically modified in such a way that they (a) no longer use the most likely words (parameter *temperature*), (b) are penalized for certain conspicuous content (*reinforcement learning*), or (c) rephrase sentences so slightly that they remain the same in terms of content but can no longer be recognized as machine-generated (*paraphrasing*). In short, detectors can easily be bypassed. The implications for society and research are discussed. Further research is needed to investigate implications, such as the influence on opinion or fake news in social media.

1. Introduction

As large language models (LLMs) continue to evolve, the necessity for precise differentiation between human outputs and those produced by LLMs is becoming increasingly critical. Recent developments in LLMs have significantly improved, particularly in complex reasoning tasks, such as mathematical problem-solving. These advancements are rapidly closing the performance gap between LLMs and human capabilities, which had previously been a major limitation of such models. Due to this close-up, the difficulty of reliably detecting LLMs has further increased.

© The Author(s), 2025. Published by Cambridge University Press. This is an Open Access article, distributed under the terms of the Creative Commons Attribution licence (http://creativecommons.org/licenses/by/4.0), which permits unrestricted re-use, distribution and reproduction, provided the original article is properly cited.

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

Various concepts helped LLMs to catch up to human-like reasoning capabilities. For example, the *Quiet-STaR* approach (Zelikman et al., 2024) reinforces intermediate beliefs generated by the model before providing a final answer, improving reasoning accuracy. The *Agent Q* framework (Putta et al., 2024) combines *Monte Carlo Tree Search* (MCTS) and *Direct Preference Optimization* (DPO) to teach LLMs to perform complex tasks, such as navigating an online store. Other work, including *Let's Verify Step by Step* (Lightman et al., 2024), enhances LLM reasoning performance by breaking tasks into discrete steps and providing feedback for each step. Likewise, *Verification for Self-Taught Reasoners* (V-STaR; Hosseini et al., 2024) refers to a concept in which the LLM generates multiple solutions for a task, learning from correct answers. A verifier model learns from correct and incorrect responses, improving the LLM's reasoning ability, particularly in coding and mathematical tasks.

In light of these rapid advancements, a growing concern is that distinguishing between human-generated text and LLM-generated text may become even more challenging. Models such as DetectGPT (Mitchell et al., 2023) and datasets like the *Human ChatGPT Comparison Corpus* (HC3; Guo et al., 2023) aim to address this challenge. However, further investigation is required to assess the reliability of these detection models and explore whether they can be circumvented with reasonable effort. Like cryptography, every detection method, as shown in the current study, is at risk of being attacked and eventually circumvented. Such insights lead to the need for additional and robust methods, such as watermarks, to help clarify the origin of published texts.

This study presents an experimental series to evaluate the reliability of LLM detection models and explore potential methods for bypassing them. In the first experiment (cf. Section 3), shallow learning classifiers are evaluated based on a Bag-of-Words (BoW) approach combined with a Naive Bayes classifier. Although this method is not state-of-the-art, it serves as a benchmark to showcase the influence of hyperparameters, including temperature, sampling method, and model size on classification performance. Additionally, the classifiers' performance is compared to human judgment. Human classification often focuses on identifying unlikely words, while machine models rely on statistical patterns (Ippolito et al., 2020). According to Gehrmann et al. (2019), human judgment achieves an accuracy of only 54% without automated assistance, improving up to 72% with supporting tools. Other studies similarly report a 50% success rate in detecting GPT-3-generated text (Clark et al., 2021). Using top-p sampling at 1.0, the Naive Bayes classifier's detection rate was reduced below 60%, indicating that simple models may be bypassed without special techniques.

In the second experiment (cf. Section 4), the combination of shallow feature categories and shallow classifiers is replaced with a BERT-based classifier and corresponding transformer-based embeddings, yielding a significantly improved accuracy exceeding 90%. This forms the basis for the initial bypassing approach. In contrast to the one described by Krishna et al. (2024), paraphrasing is not used at this stage. Instead, *reinforcement learning* (RL) is employed for model training to preserve the generative model from detection. This approach builds upon the methodology outlined by Ziegler et al. (2020), which originally fine-tuned LLMs using human feedback. Special constraints are incorporated into the reward function to prevent the model from learning trivial bypassing strategies, such as adding special characters or introducing artifacts. Depending on the LLM's size, the detection rate reduces from over 90% to below 17% following the RL training. This demonstrates that once the classifier is known and accessible, the generative model can be adapted to evade it.

In the third experiment (cf. Section 5), a paraphrasing model is applied, similar to the approach introduced by Krishna et al. (2024). While Krishna et al. focused on general-purpose paraphrasing, the model presented here is specifically tailored to conceal the generative model from DetectGPT. In this regard, a new dataset is vital. Here, the original LLM output is further paraphrased multiple times. This procedure allows us to select the version least likely to be classified as LLM-generated. Inspired by Mitchell et al. (2023), each single paraphrasing iteration altered approximately 15% of the sentence. Based on the newly created dataset, a paraphrasing model is trained to hide the original language model. By applying this paraphrasing model to the output of the Qwen1.5-4B-Chat model (Bai et al., 2023), the detection rate was iteratively reduced from 88.6% to 8.7%. A comparative analysis uses the DIPPER

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

model from Krishna et al. (2024). The presented approach hereby preserves a higher degree of linguistic similarity to the original text, even after multiple paraphrasing iterations.

In conclusion, this study demonstrates that RL and paraphrasing techniques can effectively bypass LLM detection classifiers. These results suggest that a classifier can easily be bypassed with sufficient knowledge. This can be achieved by fine-tuning (RL) or paraphrasing. The findings demonstrate the potential for malicious actors to circumvent classification. Further, the need for ongoing research into more robust and adaptive detection mechanisms is underlined.

2. Related work

Since this work combines different fields, this section is subdivided as follows: Firstly, methods for automatically generating text using LLMs are explained (Section 2.1). Secondly, relevant datasets and benchmarks are introduced (Section 2.2). Thereafter, detection methods for identifying LLM-generated content are introduced (Section 2.3). Additionally, possible ways to bypass these classifiers are discussed.

2.1. Automatic text generation

Various approaches have been employed in developing language models capable of generating text. The most prevalent architecture is based on transformer models, which include GPT and its predecessors like GPT-Neo-125M, GPT-Neo-1.3B, GPT-Neo-2.7B (Black et al., 2021), GPT-J-6B (B Wang and Komatsuzaki 2021), OPT-125M, OPT-350M, OPT-1.3B, OPT-2.7B (Zhang et al., 2022), and GPT-2 (Radford et al., 2019). Other variants encompass Instruct-GPT (Ouyang et al., 2022), Google's T5 models (Raffel et al., 2020) as well as their Gemma series (Gemma Team et al., 2024), Metas's Llama series (Dubey et al., 2024), Mistral AI's Mistral (Jiang et al., 2023) and Mixtral (Jiang et al., 2024) series, the Qwen models (Yang et al., 2024) from Alibaba Cloud, to name just a very few.

These LLMs are often referred to as *stochastic parrots* (Bender et al., 2021) since they generate sentences by predicting the next word based on probability distributions. The selection of the token with the highest probability is known as *greedy search.* As humans do not always choose the most probable sequence of words, this purely deterministic approach does not capture the variability in human language (Holtzman et al., 2020). To introduce randomness, pure random *sampling* selects tokens proportionally to their probabilities, increasing diversity but eventually reducing coherence (Bengio et al., 2003). Intermediate approaches have been developed to balance determinism and diversity. These include *typical sampling*, which prioritizes tokens near the mode of the probability distribution, and *top-k sampling*, which restricts choices to the k most probable tokens (Fan et al., 2018; Meister et al., 2023). Alternatively, *nucleus sampling* dynamically adjusts the candidate set to include only tokens whose cumulative probability remains below a threshold p (Holtzman et al., 2020). Each method reflects trade-offs between control, coherence, and creativity.

The *temperature* parameter is closely associated with *creativity.* The parameter $\tau \in [0,2]$ modulates the sharpness of the probability distribution, where higher temperatures ($\tau > 1$) flatten the distribution, allowing for more diverse and creative outputs but increasing the risk of incoherence. Conversely, lower temperatures ($\tau < 1$) sharpen the distribution, focusing on high-probability tokens and yielding more deterministic but less creative text. Different sampling strategies might be combined with τ, tailoring text generation to specific requirements.

2.2. Datasets and benchmarks

A direct comparison of human versus machine-generated content is provided by the *Human ChatGPT Comparison Corpus* (HC3; Guo et al., 2023). This dataset consists of questions regarding various topics, each answered by an LLM versus real humans.

For question-answering research, Google's *Natural Questions* corpus (NQ; Kwiatkowski et al., 2019) serves as a benchmark. This dataset consists of real user search requests. Using CNN and Daily Mail articles, another question-answering corpus is provided by Hermann et al. (2015).

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

The *Corpus of Linguistic Acceptability* dataset (CoLA; Warstadt et al., 2019) comprises sentences that have been labeled as either grammatically acceptable or unacceptable by human annotators. It might be used for training LLMs such as DeBERTa-v3-large (He, Gao et al., 2023), serving as a classifier for evaluating linguistic acceptability.

2.3. Detection techniques

In the social media context discussed here, analyses are built on the users' *behavior* or *content.* While focusing on content specifically (i.e., posts on X, formerly Twitter), the reader is referred to the literature review by Alothali et al. (2018) for deeper insights into the behavior-based approach.

Various focus and content-independent detection methods exist. According to Orabi et al. (2020), these can be categorized as *graph-based* approaches (e.g., Daya et al., 2020), *crowdsourcing* techniques (e.g., G Wang et al., 2013), *anomaly* detection methods (e.g., Nomm and Bahsi 2018), and *machine learning* (ML) based approaches (Alothali et al., 2018). This categorization by Orabi et al. can be broken down to *machine-* versus *human-based techniques* since detection is performed either manually (e.g., bot identification) or by machines, i.e., based on ML or pure statistical anomalies.

2.3.1. Human-based detection

Human performance can be examined in experimental settings. Here, human versus machine-generated texts are presented to the participants (independent variable). The subjects, who do not know the origin of the texts, are asked to identify machine-generated ones. Based on a representative random sample, this procedure allows the estimation of general human accuracy, false alarm rate, etc. (dependent variables). These scores can not only be compared to those resulting from computer-based classifiers but can also be used to identify characteristics, i.e., what humans or machines tend to prioritize and what makes them more likely to fail.

Demonstrated that way, humans tend to prioritize the semantic coherence of the text. At the same time, machine-based detection methods emphasize statistical properties such as word probabilities and sampling schemes (Ippolito et al., 2020).

Humans seem to know how and what other humans would talk about compared to machines. The fact that machines can mimic these expressions and content was used by Jakesch et al. (2023): In their experiment, participants could not distinguish between AI-generated and human-written self-presentations misled by the AI's usage of first-person pronouns, contractions, or family topics.

It can be concluded that time and resource-intensive human-based detection does not lead to better results since humans can easily be deceived by using the above-mentioned factors. This has been confirmed by Dugan et al. (2020), who introduced a tool for assessing human detection capabilities. Their demonstration of how easily humans can be deceived underlines the importance of machine- and statistical-based detection.

2.3.2. Machine-based detection

Many text generation models leave behind specific artifacts whose occurrence is extremely unlikely compared to human text (Tay et al., 2020). Those probabilities or word frequencies can be examined using statistical methods (see below). Since a manual calculation is theoretically possible here, these methods can be considered *simple.* These are to be distinguished from those methods that require the use of ML models, such as LLMs, which are therefore considered separately.

Models using the transformer architecture can be used for language generation and detection. The MultiNLI benchmark (Williams et al., 2018) allows a performance comparison regarding detection: Here, BERT-large achieves an accuracy of 88% (Lee-Thorp et al., 2022), while RoBERTa and DeBERTa score 90.8% (Y Liu et al., 2019), and 91.1% (He et al., 2021), respectively.

However, for every generator release, BERT-based classifiers such as RoBERTa must be trained again. Therefore, zero-shot classifiers like DetectGPT become quite handy (Mitchell et al., 2023). These classifiers only require two models: A duplicate of the model to be tested and a second language model

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

introducing random permutations into the test string. However, the permutation models can not handle short inputs due to their working principle.

Paraphrasing outlines the weak spot of the discussed classifiers. AI-based paraphrasing can be detected successfully, as shown by Li et al. (2024). However, this technique focuses solely on paraphrasing detection. Thus, the model is unable to determine whether the original text was human- or machine-generated. Therefore, paraphrasing can still serve as an effective bypassing technique. For instance, this technique is used by Krishna et al. (2024) and enhanced by Sadasivan et al. (2024) using recursive paraphrasing. While Krishna et al. introduced the T5-based DIPPER as their own paraphrasing model, Sadasivan et al. used DIPPER and other existing models combined without any model training or fine-tuning. Sadasivan et al. (2024) claim their model generates paraphrases with high *text quality* and *content preservation* based on human ratings on a five-point Likert scale.

The model's output can be altered not only through a separate paraphrasing model but also through a reinforcement learning approach that directly modifies the generative model. This method is inspired by the paper *Fine-Tuning Language Models from Human Preferences* by Ziegler et al. (2020).

Another approach focuses on circumventing DetectGPT (Mitchell et al., 2023) using a paraphrasing model (Krishna et al., 2024). The countermeasure suggested by Krishna et al. detects AI-generated content by comparing it to a database of AI-generated texts. This solution has two significant drawbacks: First, the provider of this AI model must establish such a service, and second, custom fine-tuned private-run models are inaccessible (Da Silva Gameiro 2024).

When the underlying generative model is known, it can be modified and hence serve as a classifier after further training (Zellers et al., 2019). Zellers et al., who originally built up their model for fake news generation, proved that this model is also most effective in detecting their own fake news. This implies that detectors originating from the generative model itself are better at detecting artificially generated fake news than standard classifiers. Both fake news generators and fake news detectors are combined by Henrique et al. (2023) as generators and discriminators in the form of a *generative adversarial network* (GAN) to demonstrate the attack against a classification model.

2.3.3. Statistical-based detection

After ChatGPT's introduction (OpenAI, 2022), the frequency of certain words, such as *intricate*, *meticulously*, *commendable*, or *meticulous*, has changed significantly in academic literature (Gray 2024a). In particular, these words increased by 50%, and others, such as *innovatively* even by 60% compared to pre-2022 levels; for a complete list of words and statistics, see the dataset by Gray (2024b). While one of these so-called *group-1* words may have been used by chance, the multiple use of *group-1* words within a text is so unlikely that it can be considered GPT-generated. Gray (2024a) estimates 1% (60.000 articles) of all 2023 publications to be machine-generated and even predicts a further increase in 2024.

Several other rule-based models exist, specially designed to identify automatically generated texts. These rely on improbable word sequences and grammar (Cabanac and Labbé 2021) and similarity measures such as word overlap (Harada et al., 2021).

A statistical-based countermeasure is the usage of watermarks in generated texts, as suggested by Kirchenbauer et al. (2023). This approach adds a constant δ to a green list of tokens while they are autoregressively sampled. So, some words appear slightly more often than others, although they are in the right position and do not jeopardize the sentence's grammar or meaning. Due to the potential of paraphrasing, detecting machine-generated text using watermarks has limitations. Removing one-quarter of the watermark tokens can be enough to evade the detection (Kirchenbauer et al., 2023). Additionally, the generation itself infers with the sampling process of the language model by withholding certain tokens and strengthening others.

3. Experiment 1: evasion of shallow detectors

Various variables may affect the *accuracy* (ACC) of a detector, above all, the type of detector itself. For example, transformer-based detectors will generally perform better than shallow ones. Although shallow

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

detectors are less accurate, they are faster and use fewer resources, making their use in practice seem realistic (e.g., real-time detection in large social networks). Serving as a benchmark, the focus of the first experiment is thus on detectors such as Naive Bayes combined with *Bag-of-Words* (BoW).

The ACC of shallow detectors is considered the dependent variable. From the related work, it can be deduced that the ACC should be influenced by the now-explained independent variables (i.e., LLM type, resp. size, sampling strategy, resp. size, as well as temperature).

The OpenAI GPT family has demonstrated that increasing the number of model parameters leads to significant improvements in language generation. Larger models, however, are particularly effective in generating longer texts due to their extended context windows. Given that the current scope is generating short tweets, varying the model size will answer the question of whether larger models also beat smaller ones when the generated output is short.

Further, the influence of the sampling strategy on the LLM token selection process is examined. It is assumed that selecting the most probable token at each step (i.e., *greedy search*) should be the easiest to detect due to determination. Conversely, pure random sampling should be the most difficult to detect for machines. However, the texts generated in this way are unusable due to their low coherence and, hence, noticeable to humans. Consequently, these two sampling strategies only set the borders for the remaining ones, which are of true relevance. These either limit the number of high-probability tokens k (*top k sampling*), consider a cumulative probability threshold p (*nucleus p sampling*), or are based on the conditional entropy of token sequences (*typical sampling*). Although it is unclear which of these strategies will perform best, previous comparisons by Holtzman et al. (2020) imply that *nucleus sampling* should be prior. However, it is unsure whether these results are generalizable to the present context, especially in combination with the other independent variables.

From the possibility of statistical detection explained above, it can be deduced that the sample size should also have an effect in addition to the type of sampling. It is hypothesized that as the sample size increases, the representativeness is enhanced, and thus, the detector's accuracy is also improved. Conversely, the ACC decreases for smaller sample sizes.

Last but not least is the influence of temperature, which adjusts the probability distribution for selecting the next word in the sequence to be predicted. Its definition implies that higher temperatures ($\tau > 1$) lead to more unusual texts, which are more difficult for machines to detect but might be recognizable to humans due to incoherence. Conversely, a low temperature ($\tau < 1$) leads to high coherence but also determinability and is, therefore, easily recognizable by machines. Since these assumptions result from the definition of the parameter itself, a moderating effect of temperature over the other factors is assumed. In other words, the effects mentioned should be additive. At least, no indications in the literature make the interaction of a specific model-sample-temperature combination seem plausible.

3.1. Methodology

The procedure involved several steps: A dataset with real tweets was first filtered to ensure that only real tweets were involved (Section 3.1.1). This partial dataset was used to fine-tune various LLMs (Section 3.1.2). The dataset was then completed using LLM-generated synthetic "fake" tweets (Section 3.1.3), yielding from different hyperparameter combinations (Section 3.1.4). A classifier was then trained with both real and fake tweets (Section 3.1.5). Using a grid-based approach, the effect of the independent variables (i.e., hyperparameters) could be measured as a result of the classifier evaluation (Section 3.2). The multi-step procedure requires the data to be split several times to provide the different models with unseen data, as depicted in Figure 1.

3.1.1. Dataset

The dataset employed in this study comprises tweets collected between January and February 2020. Due to the noisy nature of the raw data, a comprehensive set of filtering policies was implemented to refine the dataset. The following measures secure a clean dataset in one language.

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

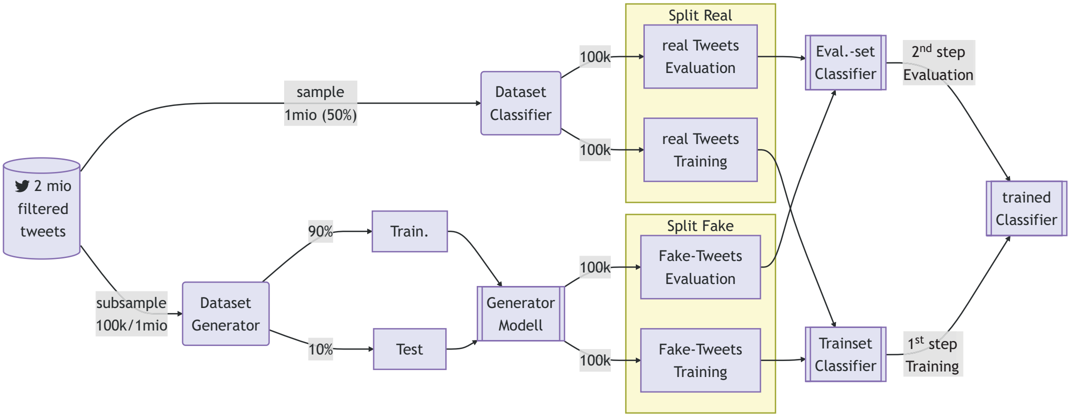

Figure 1. *Data pipeline used for modeling.*
Note: A comprehensive filtering policy was used (e.g., only tweets from verified users below the average amount of daily tweets; English language; no retweets or quoted tweets, etc.). Dataset yield is the basis for the classifier and generator, respectively. Several models were tested (e.g., pre-trained GPT versions).

In the first step, the dataset was restricted to tweets composed in English from authors with less than 100.000 followers. This measure excluded accounts from companies, sports teams, celebrities, etc., primarily used for advertising. Moreover, non-truncated tweets were selected to ensure the text was fully available, which is essential for training. To guarantee a wide variety of content, follow-up tweets, replies, quotes, and retweets were discarded. Further, only tweets from users who sent less than 20 tweets per day made it into the dataset.

After applying the mentioned filtering steps, the dataset contained approximately 2 million tweets from 136.450 verified accounts. These are set to be real tweets exclusively. For adding fake tweets to the partial dataset, it was split first for training various LLMs. The dataset was then completed using LLM-generated synthetical fake tweets. In the next step, the dataset was split up again in order to train a classifier and check its ACC accordingly.

This procedure is illustrated in Figure 1.

3.1.2. Generative models

Different open-source language models based on the GPT and OPT architecture are fine-tuned for later tweet generation. Particularly GPT-2 (Radford et al., 2019), GPT-J-6B (B Wang and Komatsuzaki 2021), GPT-Neo-125M, GPT-Neo-1.3B, and GPT-Neo-2.7B (Black et al., 2021). Furthermore, several OPT variants (125M, 350M, 1.3B, and 2.7B), as referenced in the work of Zhang et al. (2022).

3.1.3. Synthetic tweets

Using a pre-defined sampling strategy and temperature setting, each model generated two sets of 10.000 synthetic tweets for training and evaluation of the detection models, respectively. Examples are provided in Table 1.

3.1.4. Parameter grid

The independent variables are hyperparameter combinations of a parameter grid, including *temperature*, *sampling scheme*, and *-size*. *Temperature* is varied from $\tau_{min} = 0.8$ to $\tau_{max} = 1.4$ in steps of 0.2 (the results show that this part of $\tau \in [0, 2]$ is sufficient, however, for the most analysis the borders were increased to $\tau'_{min} = 0.6$, using steps of 0.1). Five sampling schemes (greedy search, typical-, top k-, nucleus- and random sampling) with four sampling sizes (1k, 10k, 50k, and 100k) were used. The whole parameter grid is used for fine-tuning a GPT-2 (1.5B) model and training the classifier, respectively.

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

Table 1. *Example tweets generated with different models*

Model	Tweet
GPT–	
Neo–125M	Kobe says new coronavirus warning on a plane is too difficult to understand.
Neo–1.3B	The new album is out now; make sure you have the album download code for free.
Neo–2.7B	#ValentinesDay: Today is the day to celebrate the greatness of yourself. And to…[1]
J–6B	“This is how we play games!” Let’s hear “The Box” tonight with @OzzyOsbourne…[1]
OPT–	
125M	I’m sure a few will be added in a future update as part of the “Duke” legacy.
350M	Good luck on the final stage of your tour!
1.3B	Rangers’ Henrik Lundqvist: “I’m not even thinking about’ the trade rumors, …”[1]
2.7B	A very cold, chilly #day for #Lincoln and #Omaha #MorningWeather

Note: Tweets generated with temperature $\tau = 1.0$ and top-50 sampling.

[1] Example trimmed due to excessive length.

A subsequent analysis is conducted to check if architecture and parameter size have an effect. The design is reduced to the most promising sampling scheme × size combination (i.e., easily detectable combinations are excluded from the design). Hence, the outstanding combination was tested for a branch of nine models with six different parameter sizes, all on the given temperature range. The used models differed in architectures (OPT vs. GPT) and parameter size, ranging from 125M to 6B.

3.1.5. Detector

A Naive Bayes classifier using BoW features was applied to detect synthetically generated tweets. This classifier has been chosen for its simplicity and short training time.

A grid-based approach was employed to train the classifier with various parameters (temperature, sampling strategy resp. size, and model architecture resp. size) all of which were explained in detail above.

3.2. Results

By definition, the temperature modification for LLM tweet generation led to modified word probability distributions accordingly. A comparison of human against machine-based quantiles revealed notable differences in temperatures of $\tau \neq 1$. This characteristic is illustrated in Figure 2a, where the frequency distribution is less visible due to logarithmic scaling. Visualized by plotting the density distribution accordingly, deviations are clearly notable for density distributions resulting from $\tau \neq 1$ (v.v. less for $\tau = 1$, cf. Figure 2b). Those depicted differences make it easy to detect variations for statistical-based classifiers, such as Naive Bayes.

Across models, sampling sizes, and temperature values, the ACC for different sampling strategies was maximal when using greedy search, as expected, and minimal when using random sampling. In the range of both strategies, nucleus sampling yielded the best results, followed by typical- and top-k sampling with similar results. For all sampling schemes, a U-shaped result pattern emerged regarding increasing temperature values, as depicted in Figure 3b. The same pattern is visible for different sampling sizes, where larger samples did not only lead to higher detection rates but also to higher gradients for temperatures diverging from 1 (cf. Figure 3a).

Across temperatures, the different models did not show a clear trend, neither in the model architecture nor in size. Particularly, both architectures lead to similar results, $M_{ACC} \approx 61.80\%$ for both types. Using Spearman’s ρ showed rather a small positive, then a negative correlation between ACC and parameter size, $\widehat{\rho} = .14$ for GPT– and $\widehat{\rho} = .13$ for OPT models (both not significant). This is visualized for different OPT– (Figure 3c) and GPT models (Figure 3d), using nucleus sampling with $p = 0.95$ and a sample size of 10k. A repetition of the experiment revealed that outliers (as shown here for OPT-2.7) were unsystematic.

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

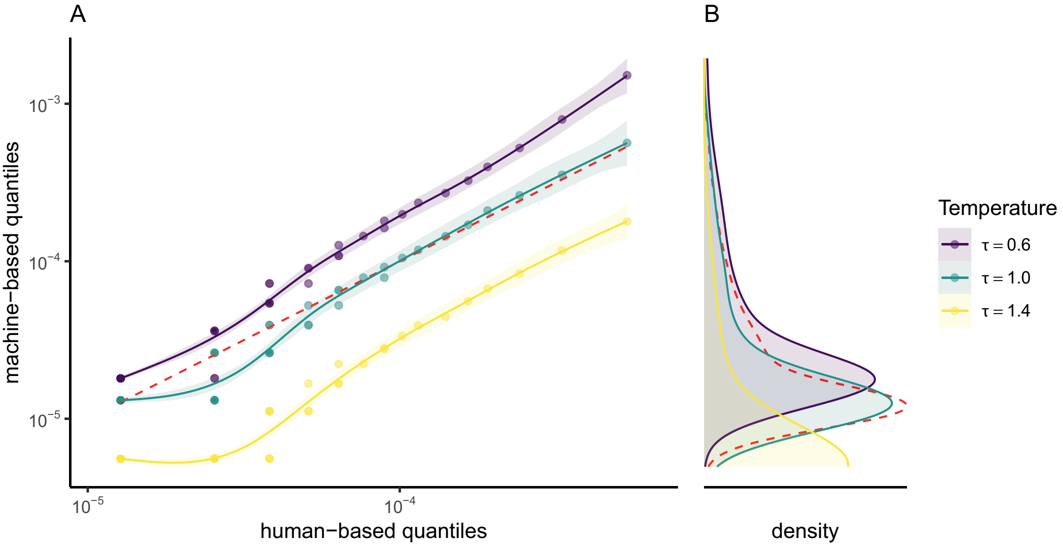

Figure 2. *Human-based against machine-based word probability distributions.*
Note: Logarithmic scaling is applied to quantiles on both axes. Machine-based quantiles result from a GPT-2 (1.5B) model using random sampling with a sampling size of 10k. (a) Comparison of machine against human-based word probability distributions (red dashed line marks theoretical perfect mapping). (b) Density distributions reflect the effect of temperature (red dashed line marks empirical human density distribution).

Notably, the same U-shaped pattern as described for sampling sizes and strategies is also visible across models: Centered at $\tau \approx 1$ where ACC is minimal, diverging temperatures lead to an increase in ACC.

3.3. Discussion

Experiment 1 focused on attacking shallow learning detectors to obtain an ACC baseline measure. Therefore, synthetic tweets were generated with the manipulation of several independent variables. The ACC of these detectors was then tested using a Naive Bayes classifier combined with BoW.

When the temperature was set to $\tau = 1$, the detection rate was minimal. Notably, there was no (curvi-) linear relationship between temperature and ACC but a U-shaped one centered around $\tau \approx 1$. At this point, human and machine-generated word probability distributions were most similar. Vice versa, $\tau \neq 1$ led to easily distinguishable distributions and, hence, to higher detection rates. This effect increased the more τ diverged from its centers. Also, the effect became stronger when sampling sizes increased (v.v. less to not visible for small samples, e.g., 1k). Regarding sampling methods, the easiest to detect was greedy search since it works by always selecting the most probable next token, introducing very little entropy (i.e., max. ACC > 99%). Comparably, the most difficult to detect was randomly sampling all possible next tokens, introducing more entropy and resulting in detection rates below 60%. However, for the purpose of coherence, using different sampling schemes is advisable. Here, nucleus sampling led to the best results (i.e., the lowest ACC). Model architecture and size, however, play a minor role in detecting short texts (i.e., similar results across architectures and no clear trend of parameter size across models). While larger models beat smaller ones when the generated output is long, this effect is not true regarding short output text. One possible explanation for this is the fact that larger models can not display their strength regarding extended context windows when texts are already small in the first place.

Taken all together, the results show, that shallow learning-based classifiers appeared to perform insufficient if the generative models produced texts with high entropy and similar word distribution to

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

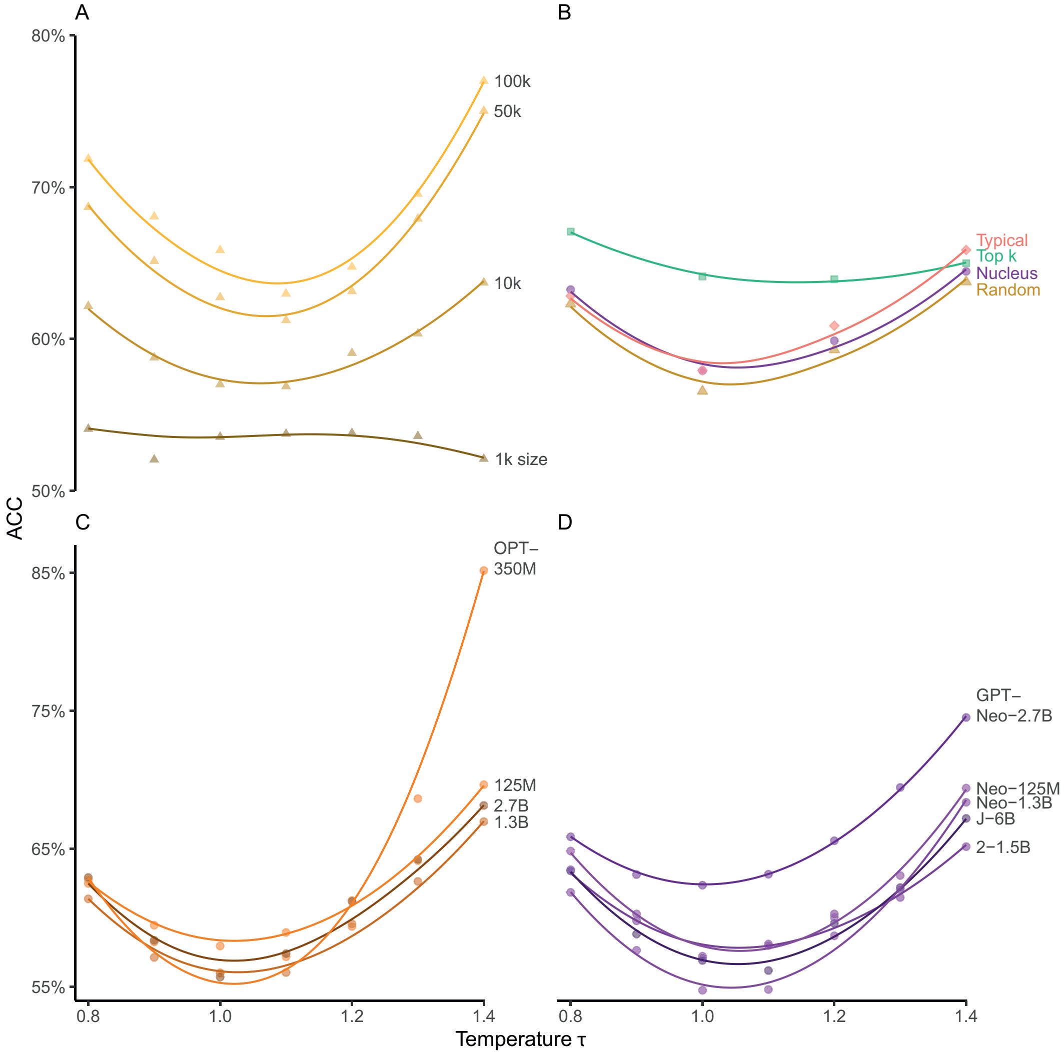

Figure 3. *Detection rates by temperature for sampling sizes, methods, and generator models.* *Note: Comparison of ACC for varying temperatures τ and (a) different sampling sizes (1k–100k tweets) using random sampling and (b) different sampling types, i.e., typical–, top* $k = 100$*, nucleus* $p = 0.95$ *and pure random sampling, all with a 10k sampling size. greedy search is not depicted here, since it always leads to ACC* $> 99\%$*. Both sampling sizes and strategies result from the same GPT-2-1.5B model, whereas panels (c) depict results for OPT– and (d) for GPT model architectures with various parameter sizes. Sampling sizes and model parameters are reflected by color shading (i.e., the darker, the bigger). For all panels, small ACC values indicate a better performance of the generating model. For colorization, see the online version of this article.*

human texts. This can be traced back to their working principle since the BoW approach does not consider the position in which a certain word is located nor synonyms with the same meaning.

Shallow detectors are resource-sparing classifiers and, therefore, represent a realistic application example (e.g., monitoring big data streams). At the same time, these detectors can be evaded easily, as demonstrated here. More advanced, resource-intensive classification models are therefore used in the field. Those models, such as those based on BERT ones, push the boundaries of the here presented

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

strategies. Hence, alternative evasion techniques are needed for transformer-based detectors, as presented in Experiment 2.

4. Experiment 2: evasion of transformer-based detectors

Given their superior performance in text classification, transformer-based models are increasingly preferred for deployment in production environments. Hence, the previously applied evasion tactics become insufficient. This raises the question of whether more advanced classification models can also be evaded.

To examine this question, the former described procedure was adopted accordingly on both sides by replacing the shallow classifier with a transformer-based one and adjusting the evasion strategy as well.

Unlike shallow learning and other deep learning algorithms, transformers make one major improvement: Their self-attention capability, formed by self-attention layers. These layers facilitate the mapping of tokens into a vector space in relation to their surrounding tokens, resulting in more contextualized representations.

Simple parameter tuning is insufficient to bypass these attention-equipped transformer models. Hence, it is vital to use more sophisticated methods. One of those might be reinforcement learning (RL). By using RL, a model can be guided towards creating a desired output. For example, it is possible to change the sentiment of a text from *negative* to *positive.* Hence, the desired output can also be a non-detectable text. RL is, therefore, used here to bypass transformers.

However, reinforcement learning's unpredictable characteristic in finding strategies for reward maximization comes with the risk of the model learning introducing artifacts to bypass the classifier. To mitigate this risk, additional guardrails are applied.

4.1. Methodology

The previously used procedure was adapted to evade transformer-based detectors. Firstly, the BoW text encodings were replaced by a transformer-based classifier (Section 4.1.2). Secondly, to bypass this classifier, reinforcement learning was used (Section 4.1.3). To further stabilize the learning process, the RLs reward function was divided into (a) the classical evasion reward (Section 4.1.4) and (b) further constraints. The latter are not only vital for stabilization but also hinder the RL process to find undesirable evasion tactics (e.g., introducing artifacts such as extensive usage of special characters).

The procedure was applied to a second scenario, i.e., the generation of fake news, to demonstrate that this methodology can be adapted to other domains. In this regard, a new dataset was used (Section 4.1.1). Training and testing procedures were similar and are therefore not again described. However, the linguistic refinement filters could be simplified to one single rule. In particular, this rule ensured that generated texts had different starting phrases, thereby maintaining some level of diversity in the output.

4.1.1. Dataset

The human dataset was the same as described in the previous experiment (Section 3.1.1). To complement the dataset with synthetic tweets, a top-50 sampling with a temperature setting of $\tau = 1.0$ was used. Additionally, the number of training samples for a BERT classifier was fixed to 100k compared to the BoW classifiers.

In order to assess the generalizability of the reinforcement learning approach, a second iteration of text generation using the CNN/Daily Mail dataset from Hermann et al. (2015) was conducted.

4.1.2. Detector

BERT was used as a primary reference model within the transformer family. This is due to its architecture, which reassembles the basics of its predecessors, such as RoBERTa and DeBERTa.

Given the considerable resources required to train an entire BERT model, fitting multiple models for the sake of hyperparameter optimization was discarded.

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

4.1.3. *Reinforcement learning*

The used hyperparameters, such as learning rate, mini-batch size, choice of the optimizer, and threshold for detecting linguistic acceptability, were based on literature recommendations. For the GPT-Neo-2.7B model, the linguistic acceptability threshold was reduced from 0.4 to 0.3, as larger models are more prone to manual interventions. Furthermore, the Adam optimizer was substituted with the Lion optimizer (Chen et al., 2024), which has reportedly outperformed the former in certain scenarios. During both the reinforcement learning and evaluation phases, the same sampling method was used to ensure consistency in results.

The general RL procedure followed the three distinct stages as described by Werra et al. (2020): *rollout*, *evaluation*, and *optimization*. Since the predefined *optimization* algorithm was not changed, only the *rollout* and *evaluation* stages are described below in more detail.

Rollout. The first stage, named *rollout*, entails the generation of synthetic tweets utilizing the language models described in Section 3.1.2. In this stage, the model is provided with the beginning tokens of an original tweet and is tasked with completing the sentence. Sometimes, the model generates entire tweets independently to mitigate the risk of overfitting short text fragments.

Evaluation. During *evaluation*, the generated texts are submitted to the BERT-based classifier described in Section 4. If the classifier recognizes the text as human-generated, the reinforcement learning algorithm receives a positive reward; otherwise, it receives a negative one. In this context, raw logits have been found to yield optimal performance.

Optimization. The final *optimization* stage entails the computation of the log probabilities of the tokens to compare the current language model with a reference model. This step represents a critical element within the reinforcement learning framework proposed by Ziegler et al. (2020), ensuring that the modified model does not overfit its generation process.

4.1.4. *Reward function*

In addition to the detector-based rewards, a carefully handcrafted reward function is introduced to further guide the text generation process. This function penalizes generated texts that, while classified as human-like, fail to meet specific linguistic criteria. The reward calculation process is illustrated in Figure 4 and consists of various rulesets, all of which will be outlined in the following paragraphs. If one or more of these linguistic rules are violated, the most severe penalty of all individual rules is applied. Conversely, should the model generate a synthetic text that satisfies all rules, the reward is equal to its evasion, as given by the detector model.

The optimization rules were developed by analyzing preliminary training runs, during which both request and response logs from the reinforcement learning process were examined. While reinforcement learning can operate without these rules, the resulting outputs are significantly less coherent. For instance, an unguided model may generate an output such as “Something for Administrator930 Macy’s Displays! RIP Family Members.” While the detector algorithm did not classify this text as machine-generated, its core message is clearly questionable. Using the ruleset below, the restriction imposed by the linguistic acceptability rule would have prevented a positive reward from being assigned to this text.

The thresholds of the further introduced penalization rules are determined by observing the reinforcement learning logs. Looking at why a training run failed helps to iteratively build rule by rule instead of knowing all the constraints right from the beginning. Another important factor for choosing the right thresholds is the type of social media text. For example, a tweet might have more special characters and emojis than a book or newspaper text; consequently, a higher threshold is chosen for these measurements.

For larger models, the reward associated with bypassing the detector algorithm can be multiplied by a scalar to prioritize the model’s circumvention over producing grammatically or semantically refined sentences if necessary.

Additional constraints. ***Special characters.*** Texts containing more than 25% special characters are penalized through a linearly decreasing negative reward, reaching the maximum penalty of –1 if the text

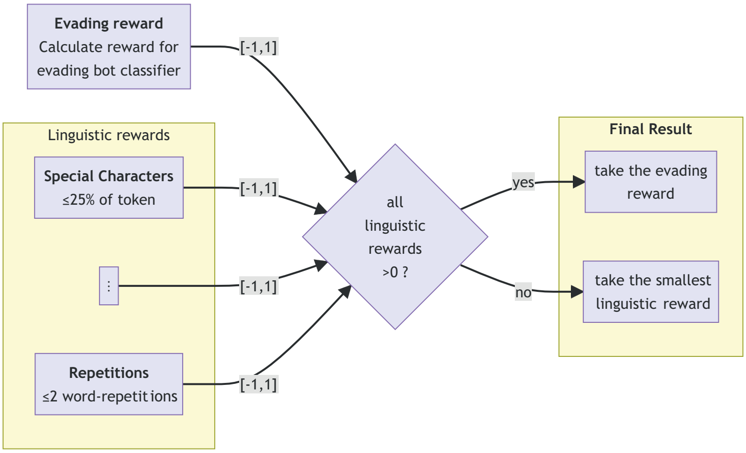

Figure 4. *Reinforcement learning reward calculation procedure.*

consists entirely of special characters. Special characters include everything except Latin letters, numbers, and white spaces, while emojis are kept out of the calculation since they are treated in an extra rule. The number of 25% is backed up by the special character to all character ratio in the trainset of the generative model, as illustrated in Figure 5a. This ratio might vary with changing text types, such as newspapers or books.

Repetitions. Besides hallucination, the repetition problem is a well-known issue in natural language generation and is therefore already scientifically analyzed (Fu et al., 2021). The idea behind the repetition penalty is to prevent the model from adopting this undesired behavior during the reinforcement learning phase. A text containing three or more instances of the same word is assigned a negative reward of up to –1 when the token is repeated eight times or more. In order not to prevent a natural text from being generated by this rule, the gold standard train corpus serves as a comparison where repetitions of over two times are very uncommon, as illustrated in Figure 5b.

Linguistic acceptability. Linguistic acceptability is evaluated using a DeBERTa-v3-large classifier (He et al., 2023), which has been trained on the Corpus of Linguistic Acceptability (CoLA) dataset (Warstadt et al., 2019). CoLA comprises sentences that have been labeled as either grammatically acceptable or unacceptable by human annotators. The trained model assesses the grammatical acceptability of a given sentence, and if its score falls below the 40% threshold, a negative reward is assigned. This reward again is linearly scaled and reaches a value of –1 if the acceptability score drops down to 0%. For models with more than two billion parameters, the threshold is relaxed up to 30% due to the increased training difficulty associated with larger models. The thresholds used in this evaluation were determined empirically rather than sourced from existing literature. Higher thresholds are typically recommended to enhance linguistic acceptability. However, excessively strict thresholds can impede the reinforcement learning process, as texts may be persistently classified as non-human, preventing the model from receiving positive rewards and hindering learning.

Dictionary. Besides introducing artifacts such as special characters, the RL process could also lead to words that are not part of any dictionary. Since this is not uncommon for tweets, it is important to give the model a certain amount of freedom to introduce unknown words. However, this opportunity should not be used excessively. Therefore, the ratio between total words and unknown words of the original corpus is taken for comparison as illustrated in Figure 5e. Because it rarely happens that less than 25% of the words are part of a dictionary, a generation with a lower score generates a negative reward of up to –1 if none of the words are found in the dictionary.

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

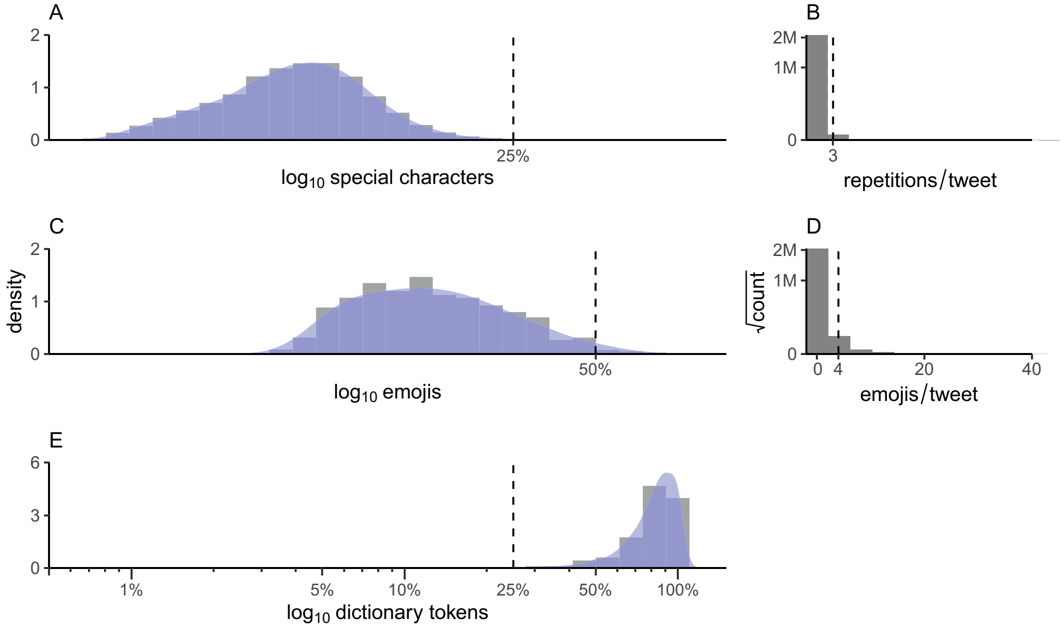

Figure 5. *Ground-truth distributions.*
Note: Constraints based on ground truth visualized for (a) the maximal proportion of special characters, (b) the number of repetitions per tweet, (c) the proportion of emojis, and (d) the number of emojis per tweet as well as (e) the minimal proportion of tokens in a standard dictionary. For all plots, the dashed line depicts the cutoff values. For all proportions (1st col.), the x-axis is log-scaled for visualization purposes, for all frequencies (2nd col.), the square root of the actual numbers is depicted on the y-axis.

Word Emoji relationship. Emojis are a common way to express emotions on social media. That's why they appear so often in social networks like X compared to newspapers or books. However, an overly excessive use of these expressions could also lead to a text losing its message and being undesirable to read. Therefore, the ground truth train dataset is once again consulted to find the maximum amount of desirable emojis within one tweet, as demonstrated in Figure 5c. The threshold of giving a negative reward is reached once a tweet contains more emojis than words (50%).

Number of Emojis. The aforementioned word-emoji relationship might not be sufficient for longer texts since, in this case, many emojis are possible. A sentence of ten words could include ten emojis, which is a bit much. This is why, additionally to the sentence length, four or more emojis are given a negative reward. Such a penalty also aligns with the natural distribution of emojis among tweets (Figure 5d).

Repetition of the Query. Although not as common as repetitions within a generated text, the repetition of the query is also a phenomenon that has been observed during RL training analysis. To conquer this flaw, repeating more than half of the query yields a negative reward.

Special Tokens. The use of special tokens, such as the beginning-of-sentence (BOS) and end-of-sentence (EOS) markers used within transformer models, is limited to two per tweet. The presence of each additional special token results in a negative reward of –0.4, with a maximum penalty of –1.

Same start. Output diversity is essential when the model generates tweets without an input query. A negative reward is imposed if more than 10% of the tweets in a training batch begin with the same word. This penalty increases linearly, reaching –1 if 20% of the tweets start similarly.

Numbers at the start. To prevent the model from learning to exploit number-based patterns to bypass the classifier, a penalty is given if generated tweets frequently start with numbers. If more than 10% of

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

tweets within a training batch exceed this limitation, the penalty is applied and scaled to a maximum once the frequency exceeds 20%.

Unknown characters. In some occasions, language models generate filler or unknown characters, typically caused by the occurrence of unknown characters included in the fine-tuning dataset. A starting penalty of –0.5 is given upon the first occurrence and decreases further for each consecutive appearance to prevent this undesirable behavior.

4.1.5. Training log

To better illustrate the internal process, Table 2 provides a sample of the ongoing process, including logs, documenting queries, responses, and rewards. As outlined in Section 4.1.3, a positive reward is only assigned when both the query and response were classified as human-generated and none of the supporting rules produced negative feedback. In these cases, the reward relies solely on the BERT classifier's ability to detect generated content.

4.2. Results

Across all evaluated BERT models, the mean F_1-score was quite high before the RL application, $M = 0.94$ ($SD = 0.01$). For the RL application, empirical analysis proved a learning rate of $5 \cdot 10^{-5}$ and a mini-batch size of 4, yielding the most optimal results. After this RL application, the mean detection rate decreased significantly to only $M = 0.09$ ($SD = 0.07$).

Further investigations regarding general applicability used the CNN/Daily Mail dataset. Despite the different domains, similar results were observed. Particularly, the detection rate decreased from $F_1 = 96\%$ to only $F_1 = 17\%$ after the RL application. Across applications, RL had a significant effect of $d = -16.48$, $95\%\mathrm{CI}[-24.14, -8.49]$. According to Sawilowsky (2009), this effect can be interpreted as *huge.* All results are summarized in Figure 6.

4.3. Discussion

Experiment 2 focused on transformer-based detection mechanisms. The 2.7 billion parameter model, with its reduced linguistic acceptability threshold, yields notably superior results, although at the cost of reduced linguistic quality. Particularly, for LLMs such as GPT2 and GPT-Neo models of different sizes (125M–2.7B), all F_1-scores exceeded 92%. This high detection rate can be explained by the enhanced capabilities in spotting textual patterns.

Although transformer-based classifiers have demonstrated high reliability in distinguishing between real and machine-generated content, a fine-tuned reinforcement learning approach can effectively bypass these robust models. The application of RL had a *huge* effect, decreasing the mean detection rates by more

Table 2. *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

Note: The RL process steps are illustrated line by line.

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

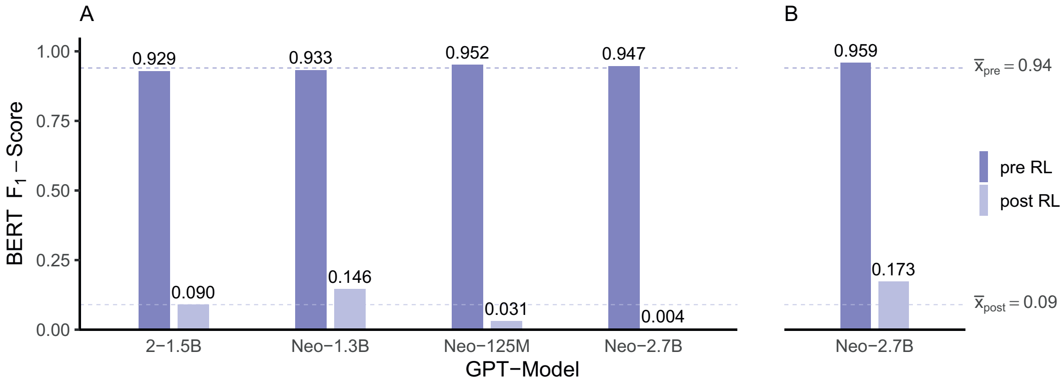

Figure 6. *F_1-score comparison.*
Note: Results are depicted for (a) the used Twitter dataset and also for (b) the CNN and Daily Mail dataset as a representation of fake news. Dashed lines depict the mean F_1-scores across models and datasets before and after RL. Line distance illustrates the huge RL effect of $d = -16.48$, $95\%\mathrm{CI}[-24.14, -8.49]$, demonstrating transferability to different text domains.

than 16 standard deviations to only $F_1 = 0.09$. This proves RL to be a reliable method for bypassing detection mechanisms. Furthermore, experiments adapted to the CNN/Daily Mail dataset demonstrate the applicability of this reinforcement learning approach to other text domains. Experiments with the four open-source models confirm that BERT classifiers can be bypassed using a reinforcement learning-based training methodology.

Despite the high detection rates, the potential for the BERT classifier to overfit to a specific generation method remains, which could lead to suboptimal performance when applied to other generative models.

5. Experiment 3: evasion of zero-shot-based detectors

Although the approaches of the first two experiments were successful, these attempts have several limitations. Primarily, the focus was solely on hiding the synthetic origin. Consequently, during RL, the model could change the content freely if it still made sense and was detected as human-written.

Additionally, the computational cost increases complementary with the parameter decrease of the model to be adjusted. This makes it increasingly difficult to run the RL approach from a hardware and stability perspective. Also, access to the original models is necessary to change them. Therefore, RL is limited to all open source models (i.e., not possible for, e.g., GPT 3 because the weights cannot be adjusted).

In order to compensate for these disadvantages and to create a general approach that is also valid for black-box models, an alternative procedure is proposed. This involves outsourcing the change to a new translation model that preserves the meaning while masking the origin. One of the many advantages of transformers was the improvement of machine translation.

The most desired goal of translations is preserving the content while remaining linguistically well-written. If transformers can be used to map from one language to another without changing the content (“translation”), it could also be possible to map from *recognizable* to *unrecognizable* with unchanged content.

For a transition to an experimental setting, maximum content similarity to the original and maximum unrecognizability are relevant. As with RL, unrecognizability is potentially in discrepancy with other influencing variables. While there is a risk of artifacts with RL, it can be assumed that content and sentence quality suffer here. Keeping the unrecognizability constantly high could lead to models, i.e., “translation” with inappropriate synonyms that are not detectable by classifiers but sound strange to humans. However, if

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

all three influencing variables specified above, including sentence quality, are taken into account, this should lead to sentences being translated (paraphrased) in such a way that their LLM origin is no longer recognizable.

5.1. Methodology

To compensate for the previous language limitations, a new dataset consisting of LLM-answered questions was created (Section 5.1.1). The answers were paraphrased, and the results were filtered to obtain the highest coherence and similarity to the original answer while being less likely to originate from an LLM (Section 5.1.3). The thereby trained paraphrasing model (Section 5.1.5) was evaluated by using a different dataset and compared to reference models (Section 5.2).

5.1.1. Dataset

This experiment utilized the *Human ChatGPT Comparison Corpus* (HC3; Guo et al., 2023) as the primary dataset. HC3 contains over 24,000 entries, each consisting of a question answered by both a human and ChatGPT. Instead of relying on the pre-existing GPT responses, new ones were generated using the *Qwen 1.5-4B* model (Bai et al., 2023). This was necessary for the subsequent step, where the same model was used to calculate log loss for permuted answers. The permutation candidates for training a paraphrasing model were generated using the *T5-3B* model without fine-tuning. Optimal permutation candidates for each answer were selected based on three criteria: *Similarity*, *coherence*, and *LLM origin plausibility* (via its log loss), as depicted in Figure 7. The masking and filtering procedures are explained in detail below. Google's Natural Questions (NQ) corpus from the Benchmark for Question Answering Research (Kwiatkowski et al., 2019) was used for evaluation purposes.

5.1.2. Sampling and masking

For each question in the HC3 dataset, a response was generated using the *Qwen 1.5-4B* model. These responses were then processed in the following manner: First, the answer was split into sentences and then tokenized using named-entity recognition (NER) to identify entities such as names, places, or numbers. In order to maintain the core message of a sentence, these tokens were preserved from substitution. For the remaining tokens, with the exception of the final token in each sentence, all possible 2-tuples of token combinations were generated for masking purposes. Each sentence could include multiple masks, as long

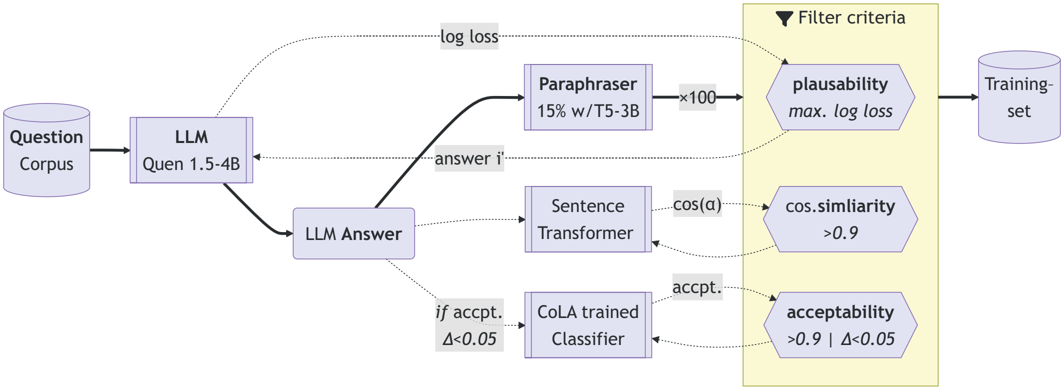

Figure 7. *Pipeline used for training set generation.*

Note: Question from Human ChatGPT Comparison Corpus (Guo et al., 2023) answered by Qwen1.5-4B-Chat (Bai et al., 2023) and permutated by t5-3b (Raffel et al., 2020). The Log loss (plausibility) is checked by the generative Qwen-model itself, while the similarity is checked using the sentence transformer all-MiniLM-L6-v2 (Wang et al., 2020). The acceptability was checked by a DeBERTa (He et al., 2021) model trained on the CoLA dataset (Warstadt et al., 2019).

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

as the masked portion did not exceed 15% of the sentence. From all possible combinations for each sentence, 10 were randomly drawn. Note that this number can also be chosen higher but was kept small in order to reduce the computational effort later on. The masked tokens were then filled using the T5 model (3B; Alisetti 2020) by generating ten paraphrased sentences per masked combination. Duplicates were discarded, ensuring variability in the output. This approach follows a methodology similar to the one described by Mitchell et al. (2023).

5.1.3. Filtering criteria

The filtering criteria were designed to select paraphrases that show the lowest likelihood of being generated by the corresponding large language model (LLM), consequently being difficult to detect as machine-generated. However, these paraphrases were also required to maintain a high degree of similarity to the original text in order to prevent semantic drift (i.e., a high similarity score is desirable). Furthermore, coherence was a crucial factor in ensuring that the paraphrases remained linguistically correct. Sentences with high similarity but incoherent or linguistically incorrect structures may evade detection by LLM classifiers due to unusual word choices. Yet, they are clearly distinguishable from human responses, making them unsuitable for the intended use. Paraphrases not meeting the mentioned criteria were discarded, while the best of the remaining ones were chosen.

Similarity. The assessment of semantic similarity was conducted using the all-MiniLM-L6-v2 model (Wang et al., 2020), which measures the cosine similarity between sentence vectors. This approach ensured that the selected paraphrases did not deviate excessively from the meaning of the original sentences, thus preserving the semantic content required for reliable training data.

Coherence. Coherence or linguistic acceptability was evaluated through a DeBERTa classifier (He et al., 2021), trained on the *Corpus of Linguistic Acceptability* dataset (CoLA; Warstadt et al., 2019). This ensured that the paraphrased sentences retained grammatical correctness and avoided structural errors that might result in linguistically faulty outputs.

LLM origin plausibility. The paraphrased sentences were evaluated using the model Qwen1.5-4B-Chat itself (Bai et al., 2023), to calculate the log loss. The paraphrase with the highest log loss, indicating the lowest likelihood of being generated by the model, was selected. This criterion aimed to identify paraphrases that were most likely to evade detection by the LLM classifier. As shown in Table 3, the sentence: *When you use Tor, your online activity is* ***encrypted and*** *can't be traced back to your* ***real identity*** is paraphrased with the goal to identify the combination with the highest cross-entropy loss.

5.1.4. Filtering and scoring

Initially, all generated examples with a cosine similarity of less than 0.9 were discarded. For coherence or linguistic acceptability, two rules were applied. If the original sentence had a coherence score of 0.9 or

Table 3. *Example of trainset generation*

Paraphrased sentence	Cross-entropy loss
When […] encrypted and […] real location.	4.582982
When […] encrypted and […] computer.	4.613022
When […] not visible to most websites and […] true location.	4.570329
When […] encrypted so that it […] ISP.	4.608329
When […] anonymous, and it […] address.	4.626064
When […] secure and […] actual computer.	4.691911
When […] secure and […] true location.	4.651963
When […] anonymous and […] actual home address or computer.	4.637465
When […] anonymous and […] ISP.	4.672797

Note: Original sentence: When you use Tor, your online activity is **encrypted and** can't be traced back to your **real identity**.

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

higher, any paraphrased sentences scoring below this threshold were rejected. Likewise, if the original sentence had a coherence score below 0.9, any paraphrases with a delta greater than ± 0.05 compared to the original were discarded, allowing only minor linguistic changes. This second delta rule is necessary because an original sentence may already have been written in a bad linguistic style. So, every paraphrased sentence would have been discarded. The remaining samples were then sorted by log loss, with the highest log loss sample being selected for each original sentence. This process yielded a training set comprising 24.300 paraphrases.

5.1.5. Paraphrasing model

The paraphrasing model is based on the T5-11B transformer (Raffel et al., 2020), trained for three epochs with a learning rate of $\alpha = 10^{-4}$. During training, the prompt "paraphrase:" was added to guide the model in generating paraphrases. T5 stands for Text-to-Text Transfer Transformer, and this is the reason why the model has been favored over other transformer models and architectures. It takes a task and the actual input as a combined input and produces the result as an output without additional explanation, introduction, hints, etc. Therefore, the paraphrasing task is treated similarly to a translation task, with the difference that the model is taught to introduce minor changes to the original sentences. Since the train data have been designed only to introduce small changes of about 15% of the tokens, the model can easily be applied recursively without removing too much of the actual meaning while being applied. This allows for the degree of change that should be applied to the original text to be adjusted.

5.1.6. Comparison models

In order to compare the conducted approach to existing ones in the scientific field, two methods by Krishna et al. (2024) and Sadasivan et al. (2024) are used. To make the comparison as significant as possible, the same data are used for Krishna et al. (2024). This is not feasible for Sadasivan et al. (2024) since their results have been human-rated, as explained below. Therefore, the results are converted for a better comparison.

Lexical diversity approach. Dipper by Krishna et al. (2024) uses a fine-tuned T5-11B transformer model. In a single paraphrasing step, the Dipper model takes two parameters, *Lexical Diversity* and *Order Diversity.* This means that instead of adjusting the degree of change by applying a paraphrasing model recursively, Dipper uses the parameter *Lexical Diversity.* Furthermore, Dipper is designed as a general-purpose paraphrasing model and is not fine-tuned to hide a specific model such as Qwen1.5-4b-chat (Bai et al., 2023).

Paraphrasing approach. A comparable approach was conducted by Sadasivan et al. (2024), who also used paraphrasing. Instead of training their own models for the hiding of the machine-generated text, the authors combined several pre-trained language models. Although optimized regarding detectability, the authors also report measures regarding grammar or text quality (similar to linguistic acceptability) and content preservation (comparable with cosine similarity). Both were human-rated on a Likert scale, thus making them less comparable.

5.2. Results

Without the application of any hiding model, the baseline detection rate was 88.9%, and the linguistic acceptability was 74.1%. Cosine similarity was, of course, 100%.

After applying the here proposed trained hiding model, all values decreased. The detection rate went down quite fast and reached 8.7% after 10 permutations. At the same time, both remaining variables were still quite high: Linguistic acceptability reached 65.5% while remaining 82.6% similarity to the original sentence. Both variables showed low variance across permutations, with mean values of $M = 68.2\%$ ($SD = 2.3\%$) regarding acceptability and $M = 89.0\%$ ($SD = 4.5\%$) for similarity.

Starting with the same baseline values, the comparing model of Krishna et al. (2024) leads to a similar linguistic acceptability of 72.5%. The detection rate decreased to 15.4%, and the cosine similarity also

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

Table 4. *Example of original answer vs. paraphrases to the question: Who was Michelangelo?*

i	Response	cos
0	Michelangelo was a very famous painter and sculptor from a long time ago. He was born in Italy and he made many beautiful pictures and statues that people still love today. Some of his most famous works include the sculpture of David and the painting of the Sistine Chapel ceiling.	1.00
1	Michelangelo was a very talented painter and sculptor who lived a long time ago. He was born in Italy and he made many × pictures and statues which are known and loved today. Some of his more famous works include the sculpture of David and the painting of the Sistine Chapel ceiling.	0.98
2	Michelangelo was a very talented painter and sculptor that became popular a long time ago. He was born in Italy, where; he made many × pictures and statues that are famous and loved today. Some of his more famous works include the sculpture of David and the painting of the Sistine Chapel.	0.97
3	Michelangelo was a very talented painter and sculptor who lived a long time ago. He was born in Italy, where he made many × paintings and statues that are well-known even today. Some of his best work includes the sculpture of David and the painting of the Sistine Chapel.	0.95
4	Michelangelo was a very talented painter and sculptor who lived during the Renaissance. He was born in Italy × and he made many × paintings and statues that are also well known even today. Some of his best work includes the sculpture of David and the paintings in the Sistine Chapel.	0.94

Note: Original prompt: “Who was Michelangelo? Explain to me like a child.”. The original answer given by Qwen1.5-4B-Chat is marked as $i = 0$ and recursive iterations as $I \geq 1$, exemplary for the first four out of ten. Colorization reflects qualitative change across applications: Light color highlights changes to the previous response, and dark color represents overtaken changes (i.e., changes compared to the original response). Deleted words or punctuation are xed out. Similarity is quantized as cosine similary (cos). For colorization, see the online version of this article.

went down, reaching 38.9%. Overall applications, mean acceptability was high with low variance $M = 72.6\%$ ($SD = 0.6\%$), contrarily to the similarity with $M = 76.4\%$ ($SD = 22.0\%$).

Hence, the detection rate is more than twice as high as with the proposed model, which did not lead to a significant decrease in similarity.

Results reported by Sadasivan et al. (2024) are less comparable since a different dataset was used. Additionally, the evaluation was done by human raters on a Likert scale regarding *grammar or text quality* (similar to *linguistic acceptability*) and *content preservation* (comparable with *cosine similarity*). Rescaling their results leads to the following values after five permutations: Acceptability of 76.8% and similarity of 67.5% were both high, but the detection rate only decreased to 60.9%. Without reporting the number of permutations, the minimal detection rate is 58.1%.

As the number of paraphrasing iterations increases, the probability of the sentence being identified as machine-generated (i.e., classifier label = 0) decreases significantly. At the same time, the likelihood of it being linguistically acceptable remains relatively stable. Furthermore, the degree of semantic similarity also decreases only marginally.

Both methods’ performance was compared by using ten iterations of the perturbation model and a lexical diversity range of 0% to 100% for the Dipper model. The benchmarks were performed on the 1,000 questions sourced from Google’s NQ corpus (Kwiatkowski et al., 2019). After six iterations of recursive paraphrasing, the detection rate dropped from 88.9% to 14.3%, while maintaining a high cosine similarity of 88% (cf. Figure 8). In contrast, achieving a comparable result with the Dipper model required setting a lexical diversity of 100%, which reduced the cosine similarity to 39%. These results demonstrate that fine-tuning a paraphrasing model to evade detection for a specific generative model can achieve superior performance compared to a general-purpose paraphrasing model.

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

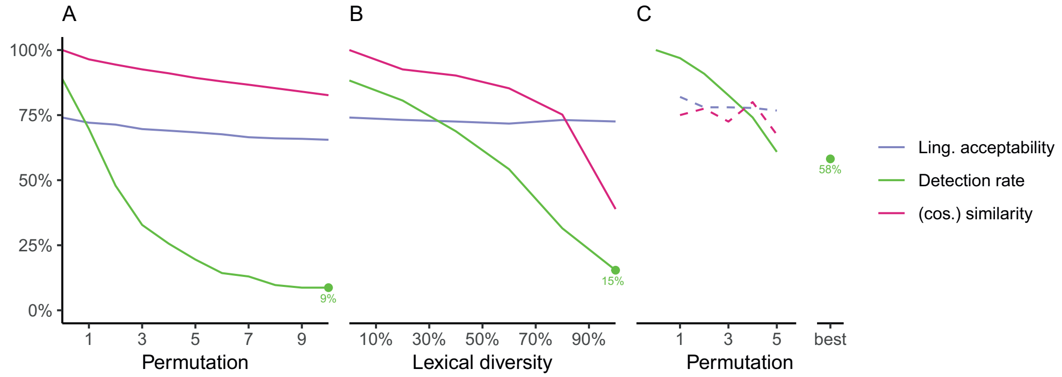

Figure 8. *Results of the proposed model vs. reference models.*
Note: Comparison of the results achieved with (a) the proposed model in comparison to (b) the Discourse Paraphraser DIPPER by Krishna et al. (2024) and (c) the paraphrasing model by Sadasivan et al. (2024). Sadasivan et al. report values for grammar or text quality (comparable to linguistic acceptability) and content preservation (matched to similarity), both manually labeled on a Likert scale of 1–5 (scaled here for better comparability but marked by dashed line representation). Those values are only provided for permutations 1–5 with detection rates, i.e., it is unclear at which permutation their "best" result occurred.

Summarized, the main advantages of the conducted new approach compared to Krishna et al. (2024) and Sadasivan et al. (2024) are a lower detectability while preserving a higher cosine similarity as well as a constant linguistic acceptability.

5.3. Discussion

The evaluation was conducted using Google's Natural Questions corpus (NQ; Kwiatkowski et al., 2019), as described in the related work. In contrast to the evaluation method proposed by Mitchell et al. (2023), this study employs a perturbation technique that modifies the responses of a language model with the objective of increasing log loss. In particular, a paraphrasing model based on the T5-11B transformer is employed to generate sentences with increased log loss. This approach enables the generation of paraphrases that progressively deviate from the original sentence, increasing its distance from the language model's typical output. Despite these modifications, the linguistic acceptability of the generated sentences is preserved.

Experiment 3 proved that post-generation paraphrasing helps to avoid detection. This is particularly useful for bigger LLMs that are harder to fine-tune or for those that are inaccessible to the user. Although paraphrasing approaches have already been proven to be applicable (Krishna et al., 2024; Sadasivan et al., 2024), it is demonstrated that the evasion results are even better if the paraphrasing model is tailored to hide one specific model. The results outperformed those of Krishna et al. by adding the recursive paraphrasing feature. The mentioned approach also surpassed Sadasivan et al, who used recursive paraphrasing, too, but combined several pre-trained paraphrasing models instead of fine-tuning them. Nevertheless, this comparison has its limitations, since Krishna et al. used a parameter for lexical diversity instead of recursively applying the model. Likewise, Sadasivan et al. used a Likert scale instead of a cosine similarity of sentence vectors to compare the permutations to the original output.

6. Conclusions and outlook

We discussed the improvement of LLMs in terms of reasoning performance, which will further complicate LLM detection in the future. In the literature review, it is noted that LLMs are no longer recognizable

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

to humans. This makes machine-based LLM detection vital. In the experimental series, it is shown that all types of state-of-the-art classifiers can be circumvented with sufficient effort. In particular, shallow detectors (Experiment 1), transformer-based detectors (Experiment 2), and zero-shot-based detectors (Experiment 3) were attacked successfully. For simple detectors, a small adjustment of the generative LLMs proved to be sufficient, such as adjusting the temperature (parameter τ in Experiment 1). However, more sophisticated detectors, as used in practice, require a better, more elaborate adjustment of the generative LLMs. During reinforcement learning, these models tend to acquire strategies that make them unrecognizable to detectors (e.g., inserting strings of special characters). While increasing evasion, these techniques tend to impede both the syntactical and semantical quality of the altered text. To counteract these model tendencies, two options were presented. Firstly, such behaviors were penalized during reinforcement learning by introducing additional constraints (cf. Experiment 2). Secondly, paraphrasing was employed, where the algorithm filtered results based on the highest similarity to the original answer, the best linguistic acceptance, and the lowest likelihood of being generated by a language model (i.e., the lowest log loss, cf. Experiment 3). With an evasion rate of $>90\%$, the model trained on the resulting dataset not only outperformed comparable models but also produced results most similar to the original response.

In the experimental series, the language used on social media, including X (formerly Twitter), was examined, and the results were generalized to internet language in general. Hence, for any current written indirect communication on the web, it is currently impossible to tell whether it is coming from a human or a machine (LLM) if the author took actions to hide the origin of the text. This everlasting battle between creating new detectors and crafting new evasion techniques will continue. Apart from the latest technical developments in this cat-and-mouse game, the question arises which implications non-detectable LLM communication will have. Of course, there are a lot of potential as well as danger. Among these is a systematic change of opinion, the spread of misinformation, or even fake news. It has been shown that people adapt their opinions to the majority, as can be seen, for example, by Asch (1956, 2016) in his still relevant conformity experiments. Although an apparently false opinion was propagated in these experiments, around a third of the test subjects adapted their own opinion to that of the other test subjects (actors) versus less than 1% without conformity pressure (control). In this regard, a transfer to social media is highly plausible: Users should hypothetically adapt their opinions to a majority of bots (posting LLM-based content) as long as those bots share the same opinion uniformly. Further research is needed for concrete proof. However, if conformity effects can be transferred to social media, it might be possible to influence the beliefs of citizens consequently. This could have an extreme social impact, such as influencing elections.

In contrast, a range of advantages exists, resulting from LLMs. For example, the replacement of low-level work, such as answering frequently asked questions repetitively. However, this also bears risks. For example, Shumailov et al. (2024a) measured the keystrokes of the agents behind the Mechanical Turk. Too few keystrokes for too long texts indicate that they must have copied their answers from ChatGPT or comparable applications instead of writing them themselves. The problem behind this practice lies in the quality of the resulting answers. When answers are designed to be used for model training but originate from LLMs in the first place, the performance of new LLMs decreases. In the long run, feeding models with model output itself leads to systematic model destruction (Shumailov et al., 2024b). This is a real problem that might become even more relevant in the future, especially if there is no way to classify input data as human-originated reliably. Already now, over 1% of academic studies are said to be generated by LLMs or with their help (Gray 2024a). This makes the affected papers useless for long-term model training. Besides the problem for future model training, Gray also denotes a potential effect on human behavior: The pure consumption of such texts might not only influence *what* we think in terms of facts, but also *how* we think to write correctly. That way, LLM text consumption might influence personal writing habits.

To support informed decision-making in applications where large language models contribute to hybrid outputs, future research should prioritize the development and implementation of tamper-resistant watermarking techniques that are compatible with diverse text generation methods. These watermarking

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

strategies allow for a clear identification of machine-generated texts while preserving the semantic integrity of the content, but are yet to be deployed more commonly. Existing approaches range from lexical and syntactical permutations over logit-based generation patterns to watermarks anchored in the LLM training procedure (Liang et al., 2024; Liu et al., 2024). From an ethical perspective, the adoption of watermarking aligns with principles of transparency, accountability, and trustworthiness.

The widespread adoption of undetectable large language models introduces moral challenges, such as the risks of manipulation of public discourse and the dissemination of biased or wrong narratives. Addressing these challenges requires a comprehensive approach to mitigation. Besides enhancing detectability via watermarking techniques, developing regulatory frameworks tailored to the ethical use of LLMs should be promoted. Further strategies may include promoting collaboration between developers, policymakers, and ethicists to establish industry-wide standards and guidelines for responsible LLM development. Finally, supporting campaigns to foster critical thinking skills among users to better evaluate the credibility of information, especially on high-volume platforms such as social media.

To conclude, large language models and their generated content bring huge opportunities to the table, but this also makes their detection more and more difficult, up to the point that they become undetectable. Society must know this risk, try to mitigate it as well as possible, and find coping strategies in case detection becomes impossible.

Data availability statement. ***Datasets:*** The *Human ChatGPT Comparison Corpus* (HC3; Guo et al., 2023) is available at Hugging Face: https://huggingface.co/datasets/Hello-SimpleAI/HC3. The *Natural Questions Corpus* (NQ; Kwiatkowski et al., 2019) of *Google's Benchmark for Question Answering Research* that was used for evaluation is also available at Hugging Face: https://huggingface.co/datasets/google-research-datasets/natural_questions. A script for generating the CNN/Daily Mail dataset (Hermann et al., 2015) is provided at GitHub: https://github.com/google-deepmind/rc-data. *Note* that the repository contains the script only and the CNN and Daily Mail articles itself have to be downloaded separately using the Wayback Machine. A processed data set is provided by NYU: https://cs.nyu.edu/~kcho/DMQA/. The Twitter data set cannot be shared because of Twitter's restrictions regarding its privacy policies. ***Models:*** The used paraphrase generator with T5 (Alisetti 2020) is available at Zenodo: https://zenodo.org/records/10731518. The fine-tuning T5-XXL *Discourse Paraphraser* (DIPPER; Krishna et al., 2024) can be downloaded at Hugging Face: https://huggingface.co/kalpeshk2011/dipper-paraphraser-xxl. The pretrained language model *Qwen 1.5-4B* (Bai et al., 2023) can also be downloaded at Hugging Face: https://huggingface.co/Qwen/Qwen1.5-4B.

Acknowledgments. The authors would like to thank the *System Sciences Chair for Communication Systems and Network Security*, as well as the audience of the *57th Hawaii International Conference* for their valuable discussions and feedback regarding the first version of this work (see Schneider et al., 2024). Special thanks also go to the CODE Research Institute for providing the hardware.

Author contributions. Funding acquisition: G.D.R.; Project administration: S.S.; Conceptualization: S.S.; Investigation: S.S., F.S., J.A.G.S.; Methodology: S.S.; Data curation & formal analysis: S.S.; Visualization: J.A.G.S.; Writing—original draft: S.S., F.S., J.A.G.S.; Writing—review & editing: S.S., F.S., J.A.G.S., G.D.R.; Supervision: G.D.R.; All authors approved the final submitted draft.

Funding statement. The authors acknowledge the financial support from the Federal Ministry of Education and Research of Germany in the program “Souverän. Digital. Vernetzt.” Joint project 6G-life, project identification number: 16KISK002.

Competing interest. None declared.

Ethical standard. The research meets all ethical guidelines, including adherence to the legal requirements of the study country.

Environment for replication purposes. All experiments were realized using a workstation equipped with four Nvidia A6000 Ada generation GPUs (48 GB VRAM each) and 512 GB of RAM. For replication purposes, a lower-level machine (e.g., only one A6000 GPU) is sufficient but requires more data generation and training time. Especially for replication of the last experiment (Experiment 3), it is strongly recommended to use the same level machine. Otherwise, a smaller model had to be chosen instead of the T5 XXL version. To train the T5 XXL model on multiple GPUs simultaneously, Microsoft's *DeepSpeed* library was used.

References

Alisetti SV (2020) Paraphrase Generator with t5 ver 1.0. Zenodo. https://doi.org/10/ndw9.

Alothali E, **Zaki N**, **Mohamed EA and Alashwal H** (2018) Detecting social bots on Twitter: A literature review. In *Proceedings of the International Conference on Innovations in Information Technology* (IIT). IEEE, pp. 175–180. https://doi.org/10/ggwcmw.

Asch SE (1956) Studies of independence and conformity: I. A minority of one against a unanimous Majority. *Psychological Monographs: General and Applied 70*(9), 1–70. https://doi.org/10/cm884h.

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

Asch SE (2016) Effects of group pressure upon the modification and distortion of judgments. In Allen RW, Porter LW and Angle HL (eds), *Organizational Influence Processes*, 2nd ed. New York: Routledge, pp. 295–303. https://doi.org/10/nkgc.

Bai J, **Bai S**, **Chu Y**, **Cui Z**, **Dang K**, **Deng X**, **Fan Y**, **Ge W**, **Han Y**, **Huang F**, **Hui B**, **Ji L**, **Li M**, **Lin J**, **Lin R**, **Liu D**, **Liu G**, **Lu C**, **Lu K**, **Ma J**, **Men R**, **Ren X**, **Ren X**, **Tan C**, **Tan S**, **Tu J**, **Wang P**, **Wang S**, **Wang W**, **Wu S**, **Xu B**, **Xu J**, **Yang A**, **Yang H**, **Yang J**, **Yang S**, **Yao Y**, **Yu B**, **Yuan H**, **Yuan Z**, **Zhang J**, **Zhang X**, **Zhang Y**, **Zhang Z**, **Zhou C**, **Zhou J**, **Zhou X and Zhu T** (2023) Qwen Technical Report. https://doi.org/10/nh4q. arXiv: 2309.16609v1 [cs.CL].

Bender EM, **Gebru T**, **McMillan-Major A and Shmitchell S** (2021) On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency* (FAccT). New York, NY, USA, pp. 610–623. https://doi.org/10/gh677h.

Bengio Y, **Ducharme R and Vincent P** (2003) A neural probabilistic Language Model. *The Journal of Machine Learning Research 3*, 1137–1155.

Black S, **Leo G**, **Wang P**, **Leahy C and Biderman S** (2021) GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow ver 1.0. Large Language Model. https://doi.org/10/kqrc.

Cabanac G and 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**, **Pham H**, **Dong X**, **Luong T**, **Hsieh CJ**, **Lu Y** et al. (2024) Symbolic discovery of optimization algorithms. In Oh A, Naumann T, Globerson A, Saenko K, Hardt M and S L (eds), *Proceedings of the 37th International Conference on Neural Information Processing Systems (NIPS)*, vol. *37*. New Orleans, LA, USA: Curran Associates Inc., 49205–49233. https://doi.org/10/kqzw.

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

Da Silva Gameiro H (2024) LLM detectors. In Kucharavy A, Plancherel O, Mulder V, Mermoud A and Lenders V (eds), *Large Language Models in Cybersecurity*. Cham, Switzerland: Springer Nature, pp. 197–204. https://doi.org/10/ndf9.

Daya AA, **Salahuddin MA**, **Limam N and 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.

Dubey A et al. (2024) The Llama 3 Herd of Models. https://doi.org/10/ndw6. arXiv: 2407.21783v2 [cs.AI] (accessed 25 August 2024).

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

Fan A, **LewisMand Dauphin Y** (2018) Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics* (ACL). vol *56*. Melbourne, Australia: Association for Computational Linguistics, pp. 889–898. Available at https://aclanthology.org/P18-1000.pdf.

Fu Z, **Lam W**, **So AMC and Shi B** (2021) A theoretical analysis of the repetition problem in text generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. *35*. 14. AAAI Press, pp. 12848–12856. https://doi.org/10/n2gm.

Gehrmann S, **Strobelt H and Rush A** (2019) GLTR: Statistical detection and visualization of generated text. In Costa-jussà MR and Alfonseca E (eds), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (ACL). Florence: Association for Computational Linguistics, pp. 111–116. https://doi.org/10/gg9p2b.

Gemma Team et al. (2024) Gemma 2: Improving open Language Models at a Practical Size. https://doi.org/10/nd57. arXiv: 2408.00118v1 [cs.CL].

Gray A (2024a) ChatGPT "Contamination": Estimating the Prevalence of LLMs in the Scholarly Literature. https://doi.org/10/mq9d. arXiv:2403.16887v1 [cs.DL].

Gray A (2024b) LLM Related Keywords in Dimensions – Search Counts. Dataset. University College London. https://doi.org/10/nfkz.

Guo B, **Zhang X**, **Wang Z**, **Jiang M**, **Nie J**, **Ding Y**, **Yue J and Wu Y** (2023) How Close Is ChatGPT to human Experts? Comparison Corpus, Evaluation, and Detection. https://doi.org/10/ndw2. arXiv: 2301.07597v1 [cs.CL] (accessed 25 August 2024).

Harada A, **Bollegala D and Chandrasiri NP** (2021) Discrimination of human-written and human and machine written sentences using text consistency. In *Proceedings of International Conference on Computing, Communication, and Intelligent Systems* (ICCCIS). New Orleans, LA, USA: Curran Associates Inc., pp. 41–47. https://doi.org/10/kq2z.

He P, **Liu X**, **Gao J and Chen W** (2021) DeBERTa: Decoding-Enhanced BERT with Disentangled Attention. https://doi.org/10/krs5. arXiv: 2006.03654v6 [cs.CL].

He P, **Gaos J and Chen W** (2023) DeBERTaV3: Improving DeBERTa Using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing. https://doi.org/10/krs4. arXiv: 2111.09543v4 [cs.CL].

Henrique DSG, **Kucharavy A and Guerraoui R** (2023) Stochastic Parrots Looking for Stochastic Parrots: LLMs Are Easy to Fine-Tune and Hard to Detect with Other LLMs. https://doi.org/10/gtg6bf. arXiv: 2304.08968 [cs.CL].

Hermann KM, **Kočiský T**, **Grefenstette E**, **Espeholt L**, **Kay W**, **Suleyman M and Blunsom P** (2015) Teaching machines to read and comprehend. In Cortes C, Lee D, Sugiyama M and Garnett R (eds), *Proceedings of the 28th International Conference on Neural Information Processing Systems* (NIPS). vol *28*. Montreal, Canada: MIT Press, pp. 1693–1701.

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

Holtzman A, **Buys Jan DL, and Forbes M and Choi Y** The curious case of neural text degeneration. Paper presented at the *Eighth International Conference on Learning Representations* (*ICLR*, April 26–May 1, 2020). Online. https://openreview.net/pdf?id=rygGQyrFvH.

Hosseini A, **Yuan X**, **Malkin N**, **Courville A**, **Sordoni A and Agarwal R** V-STaR: Training Verifiers for Self-Taught Reasoners. Paper presented at the *First Conference on Language Modeling* (*COLM*, October 7–9, 2024). Philadelphia, PA, USA. https://openreview.net/forum?id=stmqBSW2dV.

Ippolito D, **Duckworth D**, **Callison-Burch C and Eck D** (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 (ACL).* Association for Computational Linguistics, pp. 1808–1822. https://doi.org/10/knkk.

Jakesch M, **Hancock JT and Naaman M** (2023) Human heuristics for AI-generated Language are flawed. *Proceedings of the National Academy of Sciences 120*(11), e2208839120. https://doi.org/10/grwfg4.

Jiang AQ, **Sablayrolles A**, **Mensch A**, **Bamford C**, **Chaplot DS**, **Casas D de las**, **Bressand F**, **Lengyel G**, **Lample G**, **Saulnier L**, **Lavaud LR**, **Lachaux MA**, **Stock P**, **Scao TL**, **Lavril T**, **Wang T**, **Lacroix T and Sayed WE** (2023) Mistral 7B. https://doi.org/10/gtzqkt. arXiv: 2310.06825 [cs.CL].

Jiang AQ, **Sablayrolles A**, **Roux A**, **Mensch A**, **Savary B**, **Bamford C**, **Chaplot DS**, **Casas D de las**, **Hanna EB**, **Bressand F**, **Lengyel G**, **Bour G**, **Lample G**, **Lavaud LR**, **Saulnier L**, **Lachaux MA**, **Stock P**, **Subramanian S**, **Yang S**, **Antoniak S**, **Scao TL**, **Gervet T**, **Lavril T**, **Wang T**, **Lacroix T and Sayed WE** (2024) Mixtral of Experts. https://doi.org/10/gtc2g3. arXiv: 2401.04088 [cs.LG].

Kirchenbauer J, **Geiping J**, **Wen Y**, **Katz J**, **Miers I and Goldstein T** (2023) A watermark for large language models. In *Proceedings of the 40th International Conference on Machine Learning (PMLR)*, vol *202*, Honolulu, pp. 17061–17084. https://proceedings.mlr.press/v202/kirchenbauer23a.html (accessed 20 August 2024).

Krishna K, **Song Y**, **Karpinska M**, **Wieting J and Iyyer M** (2024) Paraphrasing evades detectors of AI-generated text, but retrieval is an effective defense. In Oh A, Naumann T, Globerson A, Saenko K, Hardt M and Levine S (eds), *Proceedings of the 37th International Conference on Neural Information Processing Systems (NIPS)*, vol *37*. New Orleans, LA, USA: Curran Associates Inc., pp. 27469–27500. https://dl.acm.org/doi/10.5555/3666122.3667317.

Kwiatkowski T, **Palomaki J**,**Redfield O**, **Collins M**, **Parikh A**, **Alberti C**, **Epstein D**, **Polosukhin I**, **Devlin J**, **Lee K**,**Toutanova K**, **Jones L**, **Kelcey M**, **Chang MW**, **Dai AM**, **Uszkoreit J**, **Le Q and Petrov S** (2019) Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics 7*, 453–466. https://doi.org/10/gf6gnc.

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

Li Y,**Wang Z**, **Cui L**, **Bi W**, **Shi S and Zhang Y** (2024) Spotting AI's touch: Identifying LLM-paraphrased spans in text. https://doi.org/10/ndgv. arXiv: 2405.12689 [cs.CL].

Liang Y, **Xiao J**, **Gan W and Yu PS** (2024) Watermarking Techniques for Large Language Models: A Survey. https://doi.org/10/n2f5. arXiv: 2409.00089 [cs.CR].

Lightman H, **Kosaraju V**, **Burda Y**, **Edwards H**, **Baker B**, **Lee T**, **Leike J**, **Schulman J**, **Sutskever I and Cobbe K**. Let's Verify Step by Step. Poster presented at the Twelfth International Conference on Learning Representations (*ICLR*, April 6–11, 2024). Vienna, Austria. https://iclr.cc/virtual/2024/poster/17549.

Liu Y, **Ott M**, **Goyal N**, **Du J**, **Joshi M**, **Chen D**, **Levy O**, **Lewis M**, **Zettlemoyer L and Stoyanov V** (2019) RoBERTa: A Robustly Optimized BERT Pretraining Approach. https://doi.org/10/gp5knh. arXiv: 1907.11692v1 [cs.CL].

Liu A, **Pan L**, **Lu Y**, **Li J**, **Hu X**, **Zhang X**, **Wen L**, **King I**, **Xiong H and Yu P** (2024) A Survey of Text Watermarking in the Era of Large Language Models. *ACM Computing Surveys 57*(2), Article 47. https://doi.org/10/n2fr.

Meister C, **Pimentel T**, **Wiher G and 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 CD and Finn C** (2023) DetectGPT: Zero-Shot Machine-Generated Text Detection Using Probability Curvature. https://doi.org/10/grsgh4. arXiv: 2301.11305 [cs.CL].

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

OpenAI (2022) Introducing ChatGPT. https://openai.com/index/chatgpt/ (accessed 28 September 2024).

Orabi M, **Mouheb D**, **Al Aghbari Z and 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 PF**, **Leike J and Lowe R** (2022) Training Language Models to follow Instructions with human Feedback. In Koyejo S, Mohamed S, Agarwal A, Belgrave D, Cho K and Oh A (eds), *Proceedings of the 36th International Conference on Advances in Neural Information Processing Systems (NIPS)*, vol *36*. New Orleans, LA, USA: Curran Associates, pp. 27730–27744.

Putta P, **Mills E**, **Garg N**, **Motwani S**, **Finn C**, **Garg D and Rafailov R** (2024) Agent Q: Advanced Reasoning and Learning for Autonomous AI Agents. https://doi.org/10/nhtc. arXiv: 2408.07199v1 [cs.AI].

Radford A,**Wu J**, **Child R**, **Luan D**, **Amodei D and Sutskever I** (2019) Language Models Are Unsupervised Multitask Learners. OpenAI Blog.

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press

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

Sadasivan VS, **Kumar A**, **Balasubramanian S**,**WangWand Feizi S** (2024) Can AI-Generated Text Be Reliably Detected? https://doi.org/10/gr9x8q. arXiv: 2303.11156 [cs.CL].

Sawilowsky SS (2009) New Effect Size Rules of Thumb. *Journal of Modern Applied Statistical Methods 8*(2), Article 26. https://doi.org/10/gfz8r6.

Schneider S, **Steuber F**, **Schneider JAG and Rodosek GD** (2024) How well can machine-generated texts be identified and can Language Models be trained to avoid Identification? In *Proceedings of the 57th Hawaii International Conference on System Sciences (HICSS)*, vol *57*. USA, HI, Hawaii Honolulu M¯anoa, 2716–2725. ISBN: 978-0-9981331-7-1. https://doi.org/10125/106711.

Shumailov I, **Shumaylov Z**, **Zhao Y**, **Gal Y**, **Papernot N and Anderson R** (2024a) The Curse of Recursion: Training on generated Data makes Models Forget. https://doi.org/10/kfpw. arXiv: 2305.17493v3 [cs.LG].

Shumailov I, **Shumaylov Z**, **Zhao Y**, **Papernot N**, **Anderson R and Gal Y** (2024b) AI Models collapse when trained on recursively generated Data. *Nature 631*(8022), 755–759. https://doi.org/10/gt498m.

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

Wang B and Komatsuzaki A (2021) GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. GitHub. https://github.com/kingoflolz/mesh-transformer-jax.

Wang G, **Mohanlal M**, **Wilson C**, **Wang X**, **Metzger MJ**, **Zheng H and Zhao BY** (2013) Social Turing Tests: Crowdsourcing Sybil Detection. Presentation at the *Network and Distributed System Security Symposium* (*NDSS*, February 24–27, 2013). San Diego, CA, USA. https://doi.org/10/kq3v.

Wang W, **Wei F**, **Dong L**, **Bao H**, **Yang N and Zhou M** (2020) MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. https://doi.org/10/njh6. arXiv: 2002.10957.

Warstadt A, **Singh A and Bowman SR** (2019) Neural Network Acceptability Judgments. In Lee L, Johnson M, Roark B and Nenkova A (eds), *Transactions of the Association for Computational Linguistics 7*, Stroudsburg, PA, USA: Computational Linguistics (ACL) pp. 625–641. https://doi.org/10/ggv8kd.

Werra L von, **Belkada Y**, **Tunstall L**, **Beeching E**, **Thrush T and Lambert N** (2020) TRL: Transformer Reinforcement Learning ver 0.6.0. GitHub. https://github.com/lvwerra/trl.

Williams A, **Nangia N and Bowman SR** (2018) A broad-coverage challenge corpus for sentence understanding through inference. In Walker M, Ji Hand, Stent A (eds), *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (NAACL). New Orleans: Association for Computational Linguistics, pp. 1112–1122. https://doi.org/10/gg9fnp.

Yang A, **Yang B**, **Hui B**, **Zheng B**, **Yu B**, **Zhou C**, **Li C**, **Li C**, **Liu D**, **Huang F**, **Dong G**, **Wei H**, **Lin H**, **Tang J**, **Wang J**, **Yang J**, **Tu J**, **Zhang J**, **Ma J**, **Yang J**, **Xu J**, **Zhou J**, **Bai J**, **He J**, **Lin J**, **Dang K**, **Lu K**, **Chen K**, **Yang K**, **Li M**, **Xue M**, **Ni N**, **Zhang P**, **Wang P**, **Peng R**, **Men R**, **Gao R**, **Lin R**, **Wang S**, **Bai S**, **Tan S**, **Zhu T**, **Li T**, **Liu T**, **Ge W**, **Deng X**, **Zhou X**, **Ren X**, **Zhang X**,**Wei X**, **Ren X**, **Liu X**, **Fan Y**, **Yao Y**, **Zhang Y**, **Wan Y**, **Chu Y**, **Liu Y**, **Cui Z**, **Zhang Z**, **Guo Z and Fan Z** (2024) Qwen2 Technical Report. https://doi.org/10/nkxc. arXiv: 2407.10671 [cs.CL].

Zelikman E, **Harik GR**, **Shao Y**, **Jayasiri V**, **Haber N and Goodman N** Quiet-STaR: Language models can teach themselves to think before speaking. Paper presented at the *First Conference on Language Modeling* (*COLM*, October 7–9, 2024). Philadelphia, PA, USA. https://doi.org/10/nhtb.

Zellers R, **Holtzman A**, **Rashkin H**, **Bisk Y**, **Farhadi A**, **Roesner F and Choi Y** (2019) Defending against neural fake news. In Wallach H, Larochelle H, Beygelzimer A, d'Alché-Buc F, Fox E and Garnett R (eds), *Proceedings of the 33rd International Conference on Advances in Neural Information Processing Systems* (NIPS), vol *33*. New Orleans, LA, USA: Curran Associates, Inc., pp. 9054–9065.

Zhang S, **Roller S**, **Goyal N**, **Artetxe M**, **Chen M**, **Chen S**, **Dewan C**, **Diab MT**, **Li X**, **Lin XV**, **Mihaylov T**, **Ott M**, **Shleifer S**, **Shuster K**, **Simig D**, **Koura PS**, **Sridhar A**, **Wang T and Zettlemoyer L** (2022) OPT: Open Pre-Trained Transformer Language Models. https://doi.org/10/kfxh. arXiv: 2205.01068v4 [cs.CL].

Ziegler DM, **Stiennon N**,**Wu J**, **Brown TB**, **Radford A**, **Amodei D**, **Christiano P and Irving G** (2020) Fine-Tuning Language Models from Human Preferences. https://doi.org/10/gskffn. arXiv: 1909.08593 [cs.CL].

Cite this article: Schneider S, Steuber F, Schneider JAG and Dreo Rodosek G (2025). Detection avoidance techniques for large language models. *Data & Policy*, 7: e29. doi:10.1017/dap.2025.6

https://doi.org/10.1017/dap.2025.6 Published online by Cambridge University Press