# Explainability-Based Token Replacement on LLM-Generated Text

HADI MOHAMMADI*, Department of Methodology and Statistics, Utrecht University, The Netherlands
ANASTASIA GIACHANOU, Department of Methodology and Statistics, Utrecht University, The Netherlands
DANIEL L. OBERSKI, Department of Methodology and Statistics, Utrecht University, The Netherlands
AYOUB BAGHERI, Department of Methodology and Statistics, Utrecht University, The Netherlands

**Background:** Generative models, especially large language models (LLMs), have shown remarkable progress in producing text that appears human-like. However, they often exhibit patterns that make their output easier to detect than text written by humans.
**Objectives:** In this paper, we investigate how explainable AI (XAI) methods can be used to reduce the detectability of AI-generated text (AIGT) while also introducing a robust ensemble-based detection approach.
**Methods:** We begin by training an ensemble classifier to distinguish AIGT from human-written text, then apply SHAP and LIME to identify tokens that most strongly influence its predictions. We propose four explainability-based token replacement strategies to modify these influential tokens.
**Results:** Our findings show that these token replacement approaches can significantly diminish a single classifier's ability to detect AIGT. Human evaluators achieved only 47.5% detection accuracy on rewritten texts, showing that our strategies deceive both machines and humans. However, our ensemble classifier maintains strong performance across multiple languages and domains, showing that a multi-model approach can mitigate the impact of token-level manipulations.
**Conclusions:** Our findings show that XAI methods can make AIGT harder to detect by focusing on the most influential tokens. At the same time, they highlight the need for robust, ensemble-based detection strategies that can adapt to evolving approaches for hiding AIGT.

## 1 Introduction

Recent advancements in LLMs, such as the GPT series, have enabled the creation of highly fluent text that often resembles human writing [4]. Still, even the best models often show subtle signs of being artificial, like repeating phrases, using a limited range of words, having no typos, or showing a consistent writing style. These clues can help readers or automated tools detect that the text was written by AI [35]. Being able to spot AIGT is important in various domains, such as in for protecting academic honesty, fighting misinformation, and keeping public trust. On the other hand, there are good reasons to make AI text less detectable, like tools that help people write in their own style [44], or AI summaries that aim to be especially clear and well-structured [31].

This conflict between helpful and harmful uses of LLMs creates a challenge: how to keep their benefits while preventing misuse. One proposed solution is watermarking, which adds hidden patterns to generated text to make it more transparent [17]. However, recent research shows that these watermarks can be weakened or removed by changing the text, like paraphrasing [21]. Against this backdrop, our study asks: *Can explanation methods be used to both hide and detect AI-generated text?* We contribute in two ways. First, we develop an **ensemble-based detector** that remains accurate across languages (English and Dutch) and domains (news, reviews, Twitter). Second, we show how two explanation tools, SHapley Additive exPlanations (SHAP) [24] and Local Interpretable Model-agnostic Explanations (LIME) [34], can be turned against a detector: we identify the most influential tokens and replace them via four targeted strategies (semantically similar human-preferred words,

*Corresponding Author.

Authors' Contact Information: Hadi Mohammadi, h.mohammadi@uu.nl, ORCID: 0000-0003-0860-9200, Department of Methodology and Statistics, Utrecht University, Utrecht, Utrecht, The Netherlands; Anastasia Giachanou, ORCID: 0000-0002-7601-8667, a.giachanou@uu.nl, Department of Methodology and Statistics, Utrecht University, Utrecht, Utrecht, The Netherlands; Daniel L. Oberski, ORCID: 0000-0001-7467-2297, d.l.oberski@uu.nl, Department of Methodology and Statistics, Utrecht University, Utrecht, Utrecht, The Netherlands; Ayoub Bagheri, ORCID: 0000-0001-6366-2173, a.bagheri@uu.nl, Department of Methodology and Statistics, Utrecht University, Utrecht, Utrecht, The Netherlands.

part-of-speech-constrained synonyms, GPT-based substitutions, and GPT substitutions enriched with genre context). We evaluate both *detectability* (accuracy, F1) and *textual fidelity* (BLEU, ROUGE), thereby quantifying the trade-off between hiding AI signals and preserving meaning. Our results show that token-level manipulation can fool many single-model detectors, yet a robust ensemble still performs strongly, underscoring the need to combine explainability, watermarking, and policy safeguards to manage the evolving landscape of AIGT.

The rest of this paper is organized as follows: Section 2 reviews related work on AI-generated text detection, watermarking, and adversarial rewriting. Section 3 describes the datasets and our data augmentation process. In Section 4, we present our methodology, including details on model architectures, explainability approaches, and token replacement strategies. Section 5 reports experimental results, showing how different rewriting methods affect detection performance and textual fidelity. Finally, Section 6 concludes the paper and discusses future research directions.

## 2 Related Work

Kirchenbauer et al. [17] introduced an important *red–green list* watermark. In this method, a secret hash divides the vocabulary at each token position into a preferred "green" group and a less preferred "red" group. The model then gives higher chances to the green tokens, so the generated text contains a hidden cryptographic signature. Light editing often keeps these watermarks, but bigger changes, like paraphrasing or replacing words with synonyms, can remove them [16, 21]. To make watermarks harder to notice, distortion-free methods hide information in the token-sampling *order* instead of the probability distribution. However, these can also become weak if someone rewrites the text in a tricky way [5]. Other approaches, called post-hoc schemes, add watermarks later by changing syntax templates or choosing context-aware synonyms [42], but they also have trouble surviving strong text changes.

Another important detection method takes advantage of the fact that language models (LMs) often choose very predictable words. Gehrmann et al. [12] created a token-rank histogram tool to help people find words that are "too likely" in a sentence. Zero-shot methods like *DetectGPT* look at the "curvature" of local log-probabilities; if the curvature is negative, it suggests the text was written by a machine [25]. *Fast-DetectGPT* makes this process faster and more accurate for text from ChatGPT or GPT-4 [2]. Other detection methods use features such as normalized perturbed rank [40] and the amount of unused information in token probability distributions [46]. These "white-box" methods work well when they can directly access the model's probabilities, but they become less effective without this access. To solve this problem, "DNA" regeneration methods, like DNA-GPT [43] and GPT Paternity Test [45], do not need internal access; instead, they query a black-box model again and compare the new outputs for similarity.

Stylometric approaches instead focus on word choice, sentence structure, and how the text is organized and connected. Fröhling and Zubiaga [11] grouped 35 linguistic features into four categories, low diversity, repetition, incoherence, and lack of purpose, and trained SVM and random-forest models. More advanced work introduces entity-coherence graphs [23] or journalism-specific style metrics [20], which bolster detection on brief social-media texts. Though interpretable, these handcrafted methods may be narrower in scope and less transferable across domains.

A different direction uses fine-tuned neural networks to classify AI-generated text. OpenAI's initial GPT-detector trained RoBERTa on mixed GPT-2 corpora, achieving roughly 95% accuracy on in-domain data [38]. More recent detectors employ Longformer [3] or ELECTRA to handle longer contexts and detect texts from multiple generators [7]. However, these systems often struggle with out-of-distribution domains. Recent "cross-model" or contrastive ensembles, such as Ghostbuster [41] and DeTeCtive [13], compare probability distributions from multiple open-source LMs to improve robustness.

Ensemble methods combine diverse signals to mitigate the weaknesses of individual detectors. Simple voting stacks of transformer-based classifiers outperform single models in multilingual tasks [33]. For example, recent work introduces ensemble-based approaches for detecting AI-generated text in educational content, achieving a favorable false-positive trade-off in academic scenarios [32]. Our work follows this line by designing an explanation-oriented ensemble meant to withstand token-level adversarial edits.

Paraphrasing is one of the best ways to trick AI detectors. For example, Krishna et al. [19] showed that changing the wording can make DetectGPT miss almost all AI-written text. Rewriting the text many times makes it even harder to detect [35]. Some new methods, like RADAR [14] and OUTFOX [18], are better at catching these changes because they are trained on paraphrased examples, but they still only recover some accuracy. Even small mistakes, like typos, can also confuse detectors [15]. More advanced methods can handle some edits, but none can stop all tricks or rewrites.

Paraphrasing is seen as the strongest way to avoid AI text detectors: Krishna et al. [19] found that rewriting text can reduce DetectGPT's ability to catch AI-generated text from 70% to less than 5%. Rewriting the text several times in a row (called "self-rewrites") makes it even harder to detect, while still keeping the original meaning [35]. Defenses such as RADAR [14] and OUTFOX [18] become stronger by training on paraphrased texts, and can recover 10–20 $F_1$ points. Recent studies also show that even simple changes at the letter level—like typos or swapping similar-looking letters—can confuse detectors and make them much less accurate [15]. Beyond paraphrase-based evasion, reinforcement learning has emerged as a new paradigm for generating detector-resistant text. AuthorMist [8] employs RL to iteratively refine text modifications that maximize evasion success while preserving semantic content, demonstrating that adaptive adversaries can systematically exploit detector weaknesses. Although using features like ensemble cross-probability or advanced watermarking can resist some small changes, no method is completely safe from strong rewriting attacks.

Explainability has become a cornerstone of modern NLP deployments [27]. XAI tools like SHAP and LIME show which words or tokens have the biggest effect on a detector's decisions. For example, Mitrovi'c et al. [26] used LIME to highlight important words and help people review short ChatGPT answers. But attackers can use this information in the opposite way, by changing or removing the most important words. Zhou et al. [48] showed that using SHAP-based synonyms for key words can lower detector accuracy by 40–60%. In our work, we carefully study this double-sided use of explanations, testing four types of word substitutions in two languages and different types of texts.

Beyond SHAP and LIME, alternative explainability approaches have been explored for NLP tasks. SyntaxShap [1] extends SHAP by incorporating syntactic structure, assigning importance scores to phrase-level constituents rather than individual tokens, which can provide more linguistically meaningful explanations for text generation tasks. Similarly, Integrated Gradients has been applied to detect adversarial attacks on text classifiers [30], offering gradient-based attribution that traces predictions back through neural network layers. We chose SHAP and LIME for our study because they are model-agnostic and widely applicable to any classifier architecture, including our ensemble of transformers and XGBoost models. This flexibility allows practitioners to apply our token replacement framework regardless of the underlying detection model.

A natural question arises regarding the relationship between detection-based approaches and watermarking schemes. While watermarking proactively embeds identifiable patterns during text generation [17], detection methods operate post-hoc on any text regardless of its origin. These approaches are complementary: watermarking provides provenance verification for known LLM outputs, while robust detection remains necessary for texts generated without watermarks or by adversaries who actively remove them [21]. Our XAI-based approach could potentially enhance watermarking schemes by identifying which tokens carry the strongest detection signal, informing where watermarks might be most resilient to modification. This synergy between explainability, detection, and watermarking represents a promising direction for comprehensive AI text governance.

Detectors often make mistakes with texts written by people using English as a second language. Liang et al. [22] found that seven popular GPT detectors wrongly marked more than 61% of TOEFL essays by non-native English speakers as AI-generated, even though they were very accurate with essays by native speakers. Because of this bias, some teachers are moving away from strict rules and are now encouraging students to use AI tools as support, especially for those with different language backgrounds. Since LLMs can quickly create lots of convincing fake news, it is very hard for people alone to spot which content is real [47]. To improve detection in these situations, it helps to use a mix of classifiers and to add information about where the text comes from [10].

Recent use of AI chatbots for mental health support shows that people often think these systems are caring and human-like, which leads to real and meaningful conversations. But this can be risky, especially when users do not realize they are talking to an AI. Song et al. [39] found that people in severe distress sometimes seek help from chatbots and even see them as human. In the same way, Siddals et al. [37] stressed the need for stronger safety rules and clearer information in AI mental health tools. These studies show how important it is to have good detection systems and to make it clear when someone is interacting with an AI, so that chatbots are used ethically in sensitive cases. Research also shows that no single method is perfect; reliable AI-generated text detection needs to use many signals, be tested against tricky cases, and give clear reasons for its decisions. These are the main goals of our work.

## 3 Dataset

In this section, we detail the CLIN33 dataset [9] used as our primary corpus and describe how it was augmented to expand its linguistic variability. We also present the portion of the AuTexTification dataset[1] used to strengthen our ensemble models. The CLIN33 shared task corpus comprises human-written and AIGTs in English and Dutch, including three domains: news, tweets, and reviews. Each domain contains 200 samples from human authors and 200 samples from LLMs (GPT-4 and Vicuña-13B), producing a total of 1,200 texts in English and 1,200 in Dutch. The human-written news samples come from well-known media outlets; the Dutch social media posts primarily discuss policy or health topics, whereas the English ones often revolve around social or sports events. Reviews typically cover literature in Dutch and various consumer goods in English.

Although CLIN33 offers balanced classes across two languages and three genres, its size remains relatively small for training large-scale models. To address this limitation and diversify the textual patterns, we created an augmented version of the dataset. We started with the original texts and applied a series of transformations to each sample. These transformations included synonym substitution, random word swaps, random insertions, targeted deletions, variations in spelling, and back-translation[2]. Each text thus produced several new variants that preserved the key semantics but introduced new surface forms. We ultimately obtained 9,720 total samples for training, split evenly between human and AI-generated content (4,860 each) and distributed equally across the three domains and the two languages. The final augmented corpus enables more robust feature learning.

Table 1. Overview of data resources.

| | **AuTexTification (EN)** | **Augmented CLIN33** |
|---|---|---|
| **Total Samples** | 33,845 | 9,720 |
| **Domains/Genres** | Tweets, Legal, Wiki | News, Twitter, Reviews |
| **Languages** | English | English, Dutch |
| **AI Models** | BLOOM<br>GPT-3 | GPT-4<br>Vicuña-13B |
| **Human Sources** | MultiEURLEX, Amazon, XSUM, TSATC, WikiLingua | News, tweets, reviews |
| **Class Ratio** | 50.36% / 49.64 % | 50% / 50% |
| **Augmentations** | None | Yes (9) |

As an additional resource for pre-training and fine-tuning, we employed a subset of the AuTexTification dataset[3], introduced by Sarvazyan et al. [36]. While the complete dataset includes English and Spanish

[1]https://zenodo.org/records/10732813

[2]Translating from English to Dutch and then back to English

[3]https://sites.google.com/view/autextification

texts in five distinct domains, we selectively used the *English-only* portion from tweets, legal, and wiki categories. This subset contains 33,845 texts designated for binary AI-detection (Task 1), split nearly evenly between human-written (50.36%) and AI-generated (49.64%) documents. The human content is drawn from sources such as MultiEURLEX, XSUM, Amazon Reviews, TSATC, and WikiLingua, while the AI-generated portion is produced by BLOOM (1B7, 3B, 7B1) and GPT-3 (babbage, curie, text-davinci-003). Because of its class balance and domain variety, this subset of AuTexTification serves as a valuable source of pre-training examples for our ensemble. Table 1 summarizes the essential attributes of the two datasets used to train and evaluate our methodology. The combined setup draws from the 33,845 samples in AuTexTification (English only) and the 9,720 augmented CLIN33 samples (both English and Dutch). We use AuTexTification Dataset for initial pre-training, 90% of the CLIN data for the Augmenting and training and 10% of unchanged CLIN data for evaluating.

## 4 Methodology

We used an approach that integrates several models for text classification to address the AIGT detection task. First, we converted all text to lowercase to handle tokens in a consistent way. We then removed non-informative elements such as URLs, punctuation, and special characters, since these tokens often do not carry semantic meaning. After this cleaning, we split the text into tokens and applied lemmatization to map words to their base forms, which lowered data complexity and improved the model's ability to recognize patterns.

***Baselines Models.*** We built an XGBoost classifier by first transforming the text into numerical features with a TF-IDF vectorizer set to a maximum of 5000 features. The XGBoost model was configured with use_label_encoder=False and eval_metric='logloss'. Training used the TF-IDF representations from the training set, and final predictions on the test set were evaluated using precision, recall, F1, and accuracy. Also, We fine-tuned BERT-base (bert-base-uncased), DistilBERT (distilbert-base-uncased), and XLM-RoBERTa (xlm-roberta-base). The best checkpoint was chosen based on the highest F1 score on a validation subset. We then evaluated the final models on the test data.

***Ensemble Model.*** We built an ensemble of multiple transformers instead of relying on a single one to classify human and AIGT), as inspired by Mohammadi et al. [29]. In this method, we use the strengths of each model and create an ensemble that can be fine-tuned on similar datasets. This helps the model adapt to different domains and reduces problems caused by limited data, as the features learned by each part complement each other. Specifically, we integrated bert-base-multilingual-uncased, distilbert-base-multilingual-cased,

**Algorithm 1** Combined BERT Ensemble Structure

**Input:** Text $x$
**Output:** Predicted label {*human, AI*}

**function** PretrainEnsemble($x$)
    bert_out ← $\text{BERTModel}_{\text{frozen}}(x)$
    xlmr_out ← $\text{XLMRobertaModel}_{\text{frozen}}(x)$
    distil_out ← $\text{DistilBERTModel}_{\text{frozen}}(x)$
    **return** (bert_out, xlmr_out, distil_out)
**end function**
**function** FineTuneEnsemble($x$)
    bert_out_fresh ← $\text{BERTModel}_{\text{fresh}}(x)$
    xlmr_out_fresh ← $\text{XLMRobertaModel}_{\text{fresh}}(x)$
    distil_out_fresh ← $\text{DistilBERTModel}_{\text{fresh}}(x)$
    **return** (bert_out_fresh, xlmr_out_fresh, distil_out_fresh)
**end function**
**function** CombinedModel($x$)

(froz_bert, froz_xlmr, froz_distil) ← PretrainEnsemble($x$)
    (fresh_bert, fresh_xlmr, fresh_distil) ← FineTuneEnsemble($x$)

concat_outs ← concatenate[froz_bert, froz_xlmr, froz_distil, fresh_bert, fresh_xlmr, fresh_distil]
    dense_out ← DenseLayer(concat_outs) # # L2-regularized dense layer
    $\hat{y} \leftarrow \sigma$(dense_out)  # # Probability of AI-generated text
    **if** $\hat{y} \geq 0.5$ **then**
        **return** *AI*
    **else**
        **return** *human*
    **end if**
**end function**

and xlm-roberta-base. Each transformer produced a representation of the input text, and we concatenated these representations before passing them through a dense layer with L2 regularization for final classification.

We first built a *Combined BERT* model by training three BERT models on the AuTexTification training set. We optimized the network's hyperparameters and evaluated performance on the AuTexTification test set, focusing on F1 and average metrics across folds. After training, we froze these model weights to retain the features they had learned and reduce the risk of overfitting. We next created a *combined model* that merges three BERT models with frozen weights (trained on AuTexTification) and three fresh BERT models with unfrozen parameters. These fresh models were fine-tuned on our newly augmented dataset Such an ensemble synergy aligns with Fraser et al. [10], who highlight the benefits of multi-model integration in detecting AI-generated text. Algorithm 1 outlines the training and process for our combined BERT ensemble. During training, we minimize the binary cross-entropy loss on each branch using the augmented data. We then unify the outputs in a final fully connected layer, optionally employing majority voting if multiple final heads are used. Figure 1 shows the architecture, with frozen models on the left (gray) and fresh models on the right (green).

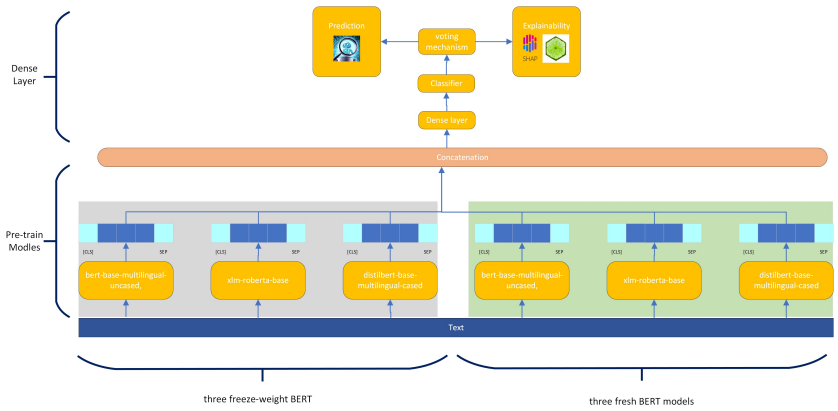


Fig. 1. The architecture of the proposed ensemble model. Outputs of three frozen BERT models (left) and three fresh BERT models (right) are concatenated and passed through dense layers, followed by a voting mechanism.

## 4.1 Experimental Setup

We trained our models using the Hugging Face Transformers library[4] and set a maximum token length of 512. Hyperparameter tuning was done through a random search over learning rates (ranging from $1 \times 10^{-5}$ to $1 \times 10^{-4}$) and batch sizes of 32, 64, or 128. We used a cosine decay schedule for learning rate adjustments, with 200 warm-up steps and an early-stopping patience of 5 epochs based on validation loss.

The final training stage used 90% of the augmented development set, while we left 10% for test. The test set comprises approximately 972 samples (486 human-written and 486 AI-generated), distributed equally across the two languages (roughly 486 English and 486 Dutch) and three domains (approximately 162 samples per domain per language). The same held-out test set was used consistently across all experiments to ensure fair comparison. The hyperparameters and early-stopping rules were adjusted using the validation split. After training, we evaluated the system on the remaining data, measuring precision, recall, F1-score, and accuracy. By combining different

[4]https://huggingface.co/

transformers, freezing selectively learned features, and training fresh models in parallel, this ensemble method balanced the advantages of robust pre-trained representations with the adaptability needed for the new dataset.

To assess statistical significance of our results, we applied bootstrap resampling with 1,000 iterations to compute 95% confidence intervals for all reported metrics. For pairwise comparisons between the Ensemble and baseline models, we employed McNemar's test at $\alpha = 0.05$, which is appropriate for comparing paired nominal predictions on the same test set. Effect sizes were calculated using Cohen's $d$ for key performance differences. In the results tables, we report confidence intervals in parentheses where space permits, and use significance markers to indicate statistical reliability: † for $p < 0.05$, $*$ for $p < 0.01$, and $**$ for $p < 0.001$.

*Computational Resources.* Table 2 summarizes the computational requirements of each model in our experiments. All training was conducted on NVIDIA A100 GPUs (40GB memory). The ensemble model, while requiring substantially more resources than individual models, remains tractable for research settings and can be parallelized across multiple GPUs to reduce wall-clock training time. For inference, the ensemble processes samples sequentially through each component, with the total inference time being the sum of individual model times plus voting overhead. In production scenarios, inference latency could be reduced through model distillation or parallel execution of the ensemble components.

Table 2. Computational cost comparison of detection models. Training times are approximate and based on the CLIN33 dataset. Inference times represent average per-sample latency on GPU.

| Model | Parameters | Training Time | Inference (ms) | GPU Memory |
|---|---|---|---|---|
| XGBoost | ~50K | ~5 min | ~1 | CPU only |
| BERT-base | 110M | ~2 h | ~50 | ~4 GB |
| DistilBERT | 66M | ~1.5 h | ~30 | ~2 GB |
| XLM-RoBERTa | 278M | ~3 h | ~60 | ~6 GB |
| Ensemble (ours) | ~660M | ~8 h total | ~200 | ~16 GB |

## 4.2 Explainability-driven detection

In this section, we identify the words that most strongly influence an *AI-generated* prediction. Our approach uses two instance-level explanation tools: SHAP and LIME. Although they both operate on individual examples, SHAP uses game-theoretic principles to estimate a token's overall (global) influence, while LIME explores small local perturbations to approximate importance. We take the absolute values of these importance scores $\phi_j$ (for SHAP) or coefficients (for LIME) to capture both positive and negative contributions.

We follow the methodology in [28], in this framework, we only analyze sentences that the model *correctly* classifies, ensuring the tokens we highlight have genuinely contributed to accurate decisions. We then average each token's importance within the sentence to get a consistent measure of its overall effect. This step helps us confirm that the extracted terms are meaningful indicators of the model's reasoning. Given a trained model $f$ and an instance $x$ composed of tokens $(t_1, t_2, \ldots, t_m)$, SHAP assigns a Shapley value $\phi_j$ to each token $t_j$. These values are computed by measuring how much including or excluding $t_j$ changes the model's output, across all possible subsets of tokens. Formally:

$$\phi_j = \sum_{S \subseteq \{1,\ldots,m\} \setminus \{j\}} \frac{|S|!\,(m - |S| - 1)!}{m!} \left[ f(S \cup \{j\}) - f(S) \right].$$

Here, $f(S)$ is the model's prediction when only the tokens in set $S$ are present. The weighting term ensures each token's effect is fairly distributed among all subsets [24]. LIME, on the other hand, makes small changes near the original text to learn a simpler, local model that approximates $f$. It then assigns coefficients reflecting each

token's importance for that instance [34]. Although LIME focuses more on local neighborhoods, it also provides per-token contributions based on how the model's prediction shifts when individual tokens are perturbed. We also consider a random token selection. This lets us compare how much improvement we gain by targeting the tokens that SHAP or LIME indicates are most influential. By using both a global (SHAP) and local (LIME) perspective, we can better trust that the identified tokens genuinely drive the model's decision. In our subsequent steps, we use these tokens to guide targeted replacements and measure how effectively such edits can mask AI-generated content.

## 4.3 Token Replacement Strategies

Once we identify influential tokens, we modify them using several approaches designed to make the text appear more human-like. We propose four rewriting strategies summarized in Table 3.

Table 3. Summary of token replacement strategies

| Strategy | Description |
|---|---|
| **Human Similar Word Replacement (HSR)** | A Word2Vec model is trained on all human-labeled texts in the training set to learn an embedding space that captures typical human usage. |
| **Part-of-Speech Replacement (PSR)** | HSR is refined by ensuring that part-of-speech tags match. |
| **GPT-based Replacement (GPT)** | A generative model (GPT-4o-mini) is queried using the prompt: `"Replace '{token}' with a more human-like word in this text: '{text}'"`. |
| **GPT-based Replacement with Genre Context (GPT+Genre)** | The GPT-based method is extended by including explicit genre information (e.g., *tweets* or *reviews*) in the prompt: `"Replace '{token}' in this {domain} text with a more human-like word: '{text}'"`. |

In *Strategy HSR*, each influential token is replaced with its nearest neighbor in the Word2Vec space, provided that the neighbor is frequently observed in human texts. In *Strategy PSR*, if a token $t$ is tagged as a noun, for instance, a Word2Vec neighbor that is also tagged as a noun is selected, preserving grammatical structure. In *Strategy GPT*, a replacement that appears more *human-like* is proposed based on the surrounding sentence. While the suggestions are context-sensitive, certain LLM-specific patterns may still be retained if the generative model reintroduces specific phrasing. In *Strategy GPT+Genre*, incorporating domain-specific information allows the generative model to propose replacements that align more closely with the original style. However, this added context may also amplify distinctive AI traits. Finally, after substituting the most influential tokens in each text, we assess performance along two dimensions: (1) changes in detectability and (2) fidelity to the original content.

## 4.4 Evaluation Metrics

We use a combination of detection and text-similarity metrics to evaluate how rewriting strategies affect our models' ability to detect AIGT, as well as how much the rewritten text deviates from its original form. First, once we rewrite an AIGT via HSR, PSR, GPT, or GPT+Genre. Next, we investigate how different rewriting scenarios affect AI-text detection. We adopt the adversarial rewriting perspective discussed in Fraser et al. [10], where paraphrased or partially edited texts can undermine naive detectors. We run it through each classifier to check whether the label remains *AI* or flips to *human*. We then calculate precision, recall, F1, and accuracy to capture each model's classification performance to detect AI/human texts under these modified inputs. Additionally, we compute the *flip rate*, defined as the fraction of AI texts whose predicted label changes to *human* after rewriting; a higher flip rate implies that the rewriting method is more effective in evading detection. In parallel, we calculate the degree of textual change introduced by each rewriting approach. We use BLEU, ROUGE-1, and ROUGE-L to measure the overlap between the original text and its rewritten form, with higher scores showing fewer modifications. By combining detection metrics, and text-similarity scores, we gain a clear view of how effectively each rewriting method hides AI-generated content and how much it changes the original text.

## 5 Experimental Analysis and Results

In this section, we compare five classification models (XGBoost, BERT-base, DistilBERT, XLM-RoBERTa, and our **Ensemble** approach) across various token-replacement scenarios for the detection of AIGT, Consistent with Fraser et al. [10], we anticipate that multi-domain evaluation is critical for robust results. Our main objective is to observe how different rewriting methods (HSR, PSR, GPT, GPT+Genre) combined with token-selection strategies (SHAP, LIME, Random) affect each model's ability to detect generated text.

Table 4 shows each model's performance on *unmodified* AIGT across two languages (English, Dutch) and three domains (news, reviews, and X/Twitter). As can be seen, although certain single models perform well on specific subsets (e.g., DistilBERT in English news, XGBoost in English Twitter), the **Ensemble** approach tends to achieve the best performance in *most* cases. This pattern is particularly clear in Dutch news, where the Ensemble model attains perfect scores. Even when other models have high recall or precision on certain tasks, they often fall behind on at least one metric (e.g., XLM-RoBERTa predicts too many positives, increasing recall but lowering precision). showing strong cross-language adaptability. This result agrees with Fraser et al. [10], who emphasize the importance of multi-lingual calibration for robust detection.

Table 4. Model performance by language and domain on unmodified AI text. **Bold** indicates the best result within each language and domain block.

| English Dataset | | | | | | Dutch Dataset | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Model** | **Domain** | **Acc** | **Prec** | **Rec** | **F1** | **Model** | **Domain** | **Acc** | **Prec** | **Rec** | **F1** |
| **— News —** | | | | | | **— News —** | | | | | |
| XGBoost | news | 0.89 | 0.93 | 0.84 | 0.88 | XGBoost | news | 0.85 | 0.88 | 0.88 | 0.88 |
| BERT-base | news | 0.94 | 0.97 | 0.91 | 0.94 | BERT-base | news | 0.68 | 0.79 | 0.56 | 0.66 |
| DistilBERT | news | 0.92 | 0.90 | **0.95** | 0.92 | DistilBERT | news | 0.68 | 0.73 | 0.66 | 0.69 |
| XLM-RoBERTa | news | 0.49 | 0.49 | 1.00 | 0.66 | XLM-RoBERTa | news | 0.53 | 0.53 | **1.00** | 0.69 |
| Ensemble | news | **0.95** | **1.00** | 0.90 | **0.95** | Ensemble | news | **0.99** | **0.98** | **1.00** | **0.99** |
| **— Reviews —** | | | | | | **— Reviews —** | | | | | |
| XGBoost | reviews | 0.51 | 0.57 | 0.59 | 0.58 | XGBoost | reviews | 0.53 | 0.51 | 0.46 | 0.48 |
| BERT-base | reviews | 0.50 | 0.54 | 0.62 | 0.58 | BERT-base | reviews | 0.40 | 0.35 | 0.30 | 0.32 |
| DistilBERT | reviews | 0.65 | 0.68 | 0.71 | 0.69 | DistilBERT | reviews | 0.42 | 0.38 | 0.50 | 0.43 |
| XLM-RoBERTa | reviews | 0.56 | 0.56 | **0.90** | 0.69 | XLM-RoBERTa | reviews | 0.43 | 0.43 | **0.60** | 0.48 |
| Ensemble | reviews | **0.67** | **0.71** | 0.68 | **0.70** | Ensemble | reviews | **0.60** | **0.53** | 0.44 | **0.49** |
| **— Twitter / X —** | | | | | | **— Twitter / X —** | | | | | |
| XGBoost | twitter | 0.91 | **0.92** | 0.98 | 0.95 | XGBoost | twitter | 0.80 | 0.85 | 0.78 | 0.81 |
| BERT-base | twitter | 0.90 | 0.86 | 0.98 | 0.92 | BERT-base | twitter | 0.80 | 0.85 | 0.78 | 0.81 |
| DistilBERT | twitter | 0.90 | 0.86 | 0.98 | 0.92 | DistilBERT | twitter | 0.84 | 0.86 | 0.80 | 0.83 |
| XLM-RoBERTa | twitter | 0.60 | 0.53 | **1.00** | 0.69 | XLM-RoBERTa | twitter | 0.62 | 0.60 | **0.90** | 0.72 |
| Ensemble | twitter | **0.93** | 0.86 | **1.00** | **0.93** | Ensemble | twitter | **0.87** | **0.90** | 0.85 | **0.87** |
| **English Average** | | | | | | **Dutch Average** | | | | | |
| XGBoost | avg | 0.77 | 0.81 | 0.80 | 0.80 | XGBoost | avg | 0.73 | 0.75 | 0.71 | 0.72 |
| BERT-base | avg | 0.78 | 0.79 | 0.84 | 0.81 | BERT-base | avg | 0.63 | 0.66 | 0.55 | 0.60 |
| DistilBERT | avg | 0.82 | 0.81 | 0.88 | 0.84 | DistilBERT | avg | 0.65 | 0.66 | 0.65 | 0.65 |
| XLM-RoBERTa | avg | 0.55 | 0.53 | 0.97 | 0.68 | XLM-RoBERTa | avg | 0.53 | 0.52 | 0.83 | 0.63 |
| Ensemble | avg | **0.85** | **0.86** | 0.86 | **0.86** | Ensemble | avg | **0.82** | **0.80** | 0.76 | **0.78** |

We next investigate how different rewriting scenarios affect AI-text detection. In particular, we apply four rewriting strategies and combine each of these with one of three token-selection strategies: SHAP (prioritizes globally influential tokens), LIME (locally influential tokens), or Random (tokens chosen arbitrarily). Table 5 presents performance metrics for each model under all these rewriting conditions.

Table 5. F1 score and Accuracy of each detector under every rewriting scenario. The best metric in a column is **bolded**; the largest negative drop is _ _ _, the smallest negative drop is ……. Δ = (Scenario Value − Original Value) × 100%.

| Explain | Strategy | XGBoost | | BERT-base | | DistilBERT | | XLM-RoBERTa | | Ensemble | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | **F1** (Δ) | **Acc** (Δ) | **F1** (Δ) | **Acc** (Δ) | **F1** (Δ) | **Acc** (Δ) | **F1** (Δ) | **Acc** (Δ) | **F1** (Δ) | **Acc** (Δ) |
| *NoRewriting* | Original | 0.81 | 0.80 | 0.74 | 0.75 | 0.79 | 0.77 | 0.68 | 0.54 | **0.83** | **0.83** |
| **SHAP** | HSR | 0.73 (-8%) | 0.75 (-5%) | 0.71 (-3%) | 0.72 (-3%) | 0.76 (-3%) | 0.75 (-2%) | 0.68 (0%) | 0.54 (0%) | **0.78** (-5%) | **0.80** (-3%) |
| **SHAP** | PSR | 0.78 (-3%) | 0.78 (-2%) | 0.73 (-1%) | 0.74 (-1%) | 0.78 (-1%) | 0.77 (0%) | 0.68 (0%) | 0.54 (0%) | **0.82** (-1%) | **0.82** (-1%) |
| **SHAP** | GPT | 0.79 (-2%) | 0.79 (-1%) | 0.73 (-1%) | 0.74 (-1%) | 0.77 (-2%) | 0.76 (-1%) | 0.68 (0%) | 0.55 (+1%) | **0.80** (-3%) | **0.81** (-2%) |
| **SHAP** | GPT+Genre | **0.80** (-1%) | **0.80** (0%) | 0.73 (-1%) | 0.74 (-1%) | 0.77 (-2%) | 0.76 (-1%) | 0.68 (0%) | 0.55 (+1%) | 0.79 (-4%) | 0.80 (-3%) |
| **LIME** | HSR | 0.25 (-56%) | 0.48 (-32%) | 0.56 (-18%) | 0.62 (-13%) | 0.62 (-17%) | 0.65 (-12%) | **0.67** (-1%) | 0.52 (-2%) | 0.64 (-19%) | **0.69** (-14%) |
| **LIME** | PSR | 0.56 (-25%) | 0.63 (-17%) | 0.64 (-10%) | 0.67 (-8%) | 0.71 (-8%) | 0.71 (-6%) | 0.66 (-2%) | 0.52 (-2%) | **0.73** (-10%) | **0.75** (-8%) |
| **LIME** | GPT | 0.57 (-24%) | 0.64 (-16%) | 0.73 (-1%) | 0.74 (-1%) | 0.78 (-1%) | 0.77 (0%) | 0.68 (0%) | 0.55 (+1%) | **0.83** (0%) | **0.83** (0%) |
| **LIME** | GPT+Genre | 0.60 (-21%) | 0.65 (-15%) | **0.75** (+1%) | **0.75** (0%) | 0.77 (-2%) | 0.76 (-1%) | 0.68 (0%) | 0.55 (+1%) | **0.81** (-2%) | **0.81** (-2%) |
| **Random** | HSR | 0.73 (-8%) | 0.74 (-6%) | 0.63 (-11%) | 0.67 (-8%) | 0.68 (-11%) | 0.69 (-8%) | 0.67 (-1%) | 0.53 (-1%) | **0.73** (-10%) | **0.75** (-8%) |
| **Random** | PSR | 0.76 (-5%) | 0.76 (-4%) | 0.65 (-9%) | 0.68 (-7%) | 0.74 (-5%) | 0.73 (-4%) | 0.67 (-1%) | 0.53 (-1%) | **0.78** (-5%) | **0.79** (-4%) |
| **Random** | GPT | 0.74 (-7%) | 0.75 (-5%) | 0.74 (0%) | 0.75 (0%) | 0.77 (-2%) | 0.76 (-1%) | 0.68 (0%) | 0.55 (+1%) | **0.81** (-2%) | **0.82** (-1%) |
| **Random** | GPT+Genre | 0.75 (-6%) | 0.75 (-5%) | 0.73 (-1%) | 0.74 (-1%) | **0.78** (-1%) | 0.77 (0%) | 0.68 (0%) | 0.55 (+1%) | **0.81** (-2%) | **0.81** (-2%) |

***General Trends.*** When no rewriting is applied (*NoRewriting, Original*), most models exhibit reasonably strong performance (precision and recall around 0.75–0.83), with the Ensemble reaching the highest overall F1 (0.83) and accuracy (0.83). After rewriting, simpler models often detect less, especially if the replaced tokens were the ones they used to identify AI patterns.

***Comparison of Token-Selection Approaches.*** SHAP or LIME-based replacements often make larger drops in detection metrics than Random replacements, since replacing highly influential tokens more effectively removes AI indicators. For instance, comparing LIME+HSR vs. Random+HSR, XGBoost's F1 can decrease from around 0.73 to 0.25 if LIME is used, indicating a big loss in detection ability once key tokens are perturbed. On the other hand, random token replacement leaves more of the AI signals intact, so detection remains higher.

***Influence of Rewriting Methods.*** HSR and PSR replace tokens with synonyms from a human corpus, often preserving the original text's structure. This is reflected in relatively high BLEU scores (e.g., over 90% in the SHAP+HSR or SHAP+PSR rows), meaning minimal rewriting. Nonetheless, these small edits can still mislead detectors if the replaced tokens were crucial. GPT-based rewrites typically lower BLEU into the 70s or 80s, reflecting more substantial paraphrases. In turn, many detectors (e.g., XGBoost or XLM-RoBERTa) experience an increase in recall but a decrease in precision, or the opposite, because the text shifts away from familiar AI phrasing. GPT+Genre further contextualizes the rewrite with domain hints (e.g. write it like a news article), causing domain-sensitive lexical changes. This can introduce new phrasing that confuses detectors reliant on domain-specific cues.

***Model-Specific Observations.*** XLM-RoBERTa often shows high recall (sometimes hitting 1.0), but its precision remains around 0.52, leading to only modest changes in F1. DistilBERT is more balanced yet remains vulnerable when its top keywords are replaced by synonyms or paraphrases. BERT-base typically holds a moderate advantage across scenarios but can be undermined in LIME-based rewrites (e.g., LIME+HSR, where BERT-base F1 falls to about 0.56). By contrast, the **Ensemble** approach consistently retains the highest or near-highest F1 and accuracy.

For example, under LIME+GPT, the Ensemble still attains an F1 of 0.83, significantly above XGBoost's 0.57 or BERT-base's 0.73. Similarly, in SHAP plus GPT+Genre, the Ensemble preserves a strong F1 of about 0.79, while other models vary more dramatically. Also, We show text similarity scores (BLEU, ROUGE1, ROUGEL) to capture how extensively each rewrite modifies the original text.

***Text Similarity Implications.*** Table 6 presents text similarity scores (BLEU, ROUGE1, ROUGEL) under all these rewriting conditions. Looking at BLEU and ROUGE scores confirms that GPT-based methods, especially GPT+Genre, introduce the largest textual shifts. Replacements guided by synonyms (HSR or PSR) keep BLEU near or above 90, whereas GPT rewrites can dip into the mid-70s. This trade-off highlights how more extensive rewriting can more thoroughly remove AI signatures at the expense of text fidelity.

***Flip Rates and Overlapping Flips.*** To estimate how many AIGTs are relabeled as *human* after rewriting, we track *flip rates* for each combination of explainability method, rewriting strategy, and model. A higher flip rate shows that the corresponding model is more easily deceived by that particular rewriting approach. In these experiments, LIME-based edits coupled with HSR lead to notably high misclassification rates in simpler models such as XGBoost (flipping up to 65% of the samples), whereas XLM-RoBERTa remains relatively robust at a mere 2.5% flip rate. GPT-based rewrites result in mid-level flip percentages overall; for example, under LIME+GPT, XGBoost flips 31.67% of texts while BERT-base and DistilBERT are both at around 9.17%. Moreover, SHAP+PSR frequently produces lower flip rates, showing that while part-of-speech-constrained edits can deceive less sophisticated classifiers, advanced or ensemble approaches remain relatively unaffected (with an Ensemble flip rate of only 2.5%). Notably, XLM-RoBERTa's tendency to overpredict *AI* keeps its flips artificially low. The Ensemble, although not entirely immune, stays in the lower to mid-range across all scenarios, once again showing greater robustness than single-model detectors. Table 7 provides a detailed breakdown of which sets of models flip each sample in various scenarios.

Table 6. BLEU, ROUGE-1 and ROUGE-L scores for every rewriting scenario.

| Explain | Strategy | BLEU (Δ) | ROUGE-1 (Δ) | ROUGE-L (Δ) |
|---|---|---|---|---|
| *NoRewriting* | Original | 1,00 | 1,00 | 1,00 |
| **SHAP** | HSR | 0,93 (-7%) | 0,97 (-3%) | 0,97 (-3%) |
| **SHAP** | PSR | **0,97** (-3%) | **0,99** (-1%) | **0,99** (-1%) |
| **SHAP** | GPT | 0,86 (-14%) | 0,93 (-7%) | 0,93 (-7%) |
| **SHAP** | GPT+Genre | 0,86 (-14%) | 0,94 (-6%) | 0,94 (-6%) |
| **LIME** | HSR | 0,84 (-16%) | 0,92 (-8%) | 0,92 (-8%) |
| **LIME** | PSR | 0,90 (-10%) | 0,96 (-5%) | 0,95 (-5%) |
| **LIME** | GPT | 0,76 (-24%) | 0,84 (-16%) | 0,84 (-16%) |
| **LIME** | GPT+Genre | 0,76 (-24%) | 0,85 (-15%) | 0,84 (-16%) |
| **Random** | HSR | 0,84 (-16%) | 0,92 (-8%) | 0,92 (-8%) |
| **Random** | PSR | 0,90 (-11%) | 0,95 (-5%) | 0,95 (-5%) |
| **Random** | GPT | 0,76 (-24%) | 0,84 (-16%) | 0,84 (-16%) |
| **Random** | GPT+Genre | 0,76 (-24%) | 0,84 (-16%) | 0,84 (-16%) |

As shown in Table 7, many rewriting strategies cause a lot of flips for some detectors, especially the simpler ones. However, the Ensemble model usually has a lower flip rate. This shows that using several models together makes detection more reliable.

***Interpreting Overlap Patterns.*** Figure 2 visualizes the overlap patterns among flipped samples across all models using UpSet plots. Each bar represents the number of AI-generated samples that were simultaneously flipped (misclassified as human) by a specific combination of models, with the connected dots below indicating which models are involved in that intersection.

Several patterns emerge from this analysis. First, XGBoost-only flips dominate many scenarios, particularly under LIME+HSR where XGBoost uniquely flips 40+ samples that other models correctly classify. This indicates that XGBoost's feature-based approach is most susceptible to targeted token replacements. Second, the Ensemble exhibits notably smaller exclusive flip counts compared to other models—typically 3–11 samples depending on

Table 7. Flip rates (%) from **AI** to **human**, grouped by rewriting strategies. Each scenario has 120 AI-labeled items; higher values indicate greater susceptibility to "flipping."

(a) HSR & PSR Strategies

| Explain | Strategy | Model | #AI | Flips | Flip Rate |
|---|---|---|---|---|---|
| SHAP | HSR | XGBoost | 120 | 17 | 14.17 |
| | | BERT | 120 | 7 | 5.83 |
| | | DistilBERT | 120 | 9 | 7.50 |
| | | XLM-R | 120 | 1 | 0.83 |
| | | Ensemble | 120 | 8 | 6.67 |
| | PSR | XGBoost | 120 | 7 | 5.83 |
| | | BERT | 120 | 4 | 3.33 |
| | | DistilBERT | 120 | 4 | 3.33 |
| | | XLM-R | 120 | 0 | 0.00 |
| | | Ensemble | 120 | 3 | 2.50 |
| LIME | HSR | XGBoost | 120 | 78 | 65.00 |
| | | BERT | 120 | 31 | 25.83 |
| | | DistilBERT | 120 | 32 | 26.67 |
| | | XLM-R | 120 | 3 | 2.50 |
| | | Ensemble | 120 | 33 | 27.50 |
| | PSR | XGBoost | 120 | 46 | 38.33 |
| | | BERT | 120 | 20 | 16.67 |
| | | DistilBERT | 120 | 19 | 15.83 |
| | | XLM-R | 120 | 5 | 4.17 |
| | | Ensemble | 120 | 19 | 15.83 |
| Random | HSR | XGBoost | 120 | 15 | 12.50 |
| | | BERT | 120 | 20 | 16.67 |
| | | DistilBERT | 120 | 22 | 18.33 |
| | | XLM-R | 120 | 2 | 1.67 |
| | | Ensemble | 120 | 20 | 16.67 |
| | PSR | XGBoost | 120 | 10 | 8.33 |
| | | BERT | 120 | 17 | 14.17 |
| | | DistilBERT | 120 | 11 | 9.17 |
| | | XLM-R | 120 | 1 | 0.83 |
| | | Ensemble | 120 | 14 | 11.67 |

(b) GPT & GPT+Genre Strategies

| Explain | Strategy | Model | #AI | Flips | Flip Rate |
|---|---|---|---|---|---|
| SHAP | GPT | XGBoost | 120 | 4 | 3.33 |
| | | BERT | 120 | 7 | 5.83 |
| | | DistilBERT | 120 | 7 | 5.83 |
| | | XLM-R | 120 | 0 | 0.00 |
| | | Ensemble | 120 | 10 | 8.33 |
| | GPT+Genre | XGBoost | 120 | 4 | 3.33 |
| | | BERT | 120 | 3 | 2.50 |
| | | DistilBERT | 120 | 6 | 5.00 |
| | | XLM-R | 120 | 0 | 0.00 |
| | | Ensemble | 120 | 11 | 9.17 |
| LIME | GPT | XGBoost | 120 | 38 | 31.67 |
| | | BERT | 120 | 11 | 9.17 |
| | | DistilBERT | 120 | 11 | 9.17 |
| | | XLM-R | 120 | 0 | 0.00 |
| | | Ensemble | 120 | 11 | 9.17 |
| | GPT+Genre | XGBoost | 120 | 41 | 34.17 |
| | | BERT | 120 | 4 | 3.33 |
| | | DistilBERT | 120 | 7 | 5.83 |
| | | XLM-R | 120 | 0 | 0.00 |
| | | Ensemble | 120 | 9 | 7.50 |
| Random | GPT | XGBoost | 120 | 16 | 13.33 |
| | | BERT | 120 | 6 | 5.00 |
| | | DistilBERT | 120 | 6 | 5.00 |
| | | XLM-R | 120 | 0 | 0.00 |
| | | Ensemble | 120 | 5 | 4.17 |
| | GPT+Genre | XGBoost | 120 | 13 | 10.83 |
| | | BERT | 120 | 6 | 5.00 |
| | | DistilBERT | 120 | 9 | 7.50 |
| | | XLM-R | 120 | 0 | 0.00 |
| | | Ensemble | 120 | 11 | 9.17 |

the scenario—demonstrating that samples fooling the Ensemble are usually also fooling other models, suggesting these are inherently difficult cases rather than ensemble-specific weaknesses.

Third, the overlap between XGBoost and transformer models is relatively limited, confirming that their failure modes differ substantially. This complementarity is precisely why the ensemble architecture succeeds: when one detection pathway fails, others often succeed. For instance, under SHAP+GPT, fewer than 5 samples fool all five models simultaneously, meaning the vast majority of evasion attempts fail against at least one component.

Fourth, XLM-RoBERTa shows consistently minimal participation in flip intersections due to its tendency to overpredict AI, which paradoxically makes it resistant to evasion attacks while reducing precision. The Ensemble balances this trade-off by combining XLM-RoBERTa's robustness with other models' higher precision.

These overlap patterns quantify the ensemble's robustness advantage: even when 27.5% of samples flip under LIME+HSR for the Ensemble, the shared samples with other models suggest these represent the hardest adversarial cases where extensive token modifications fundamentally alter the text's detectable characteristics

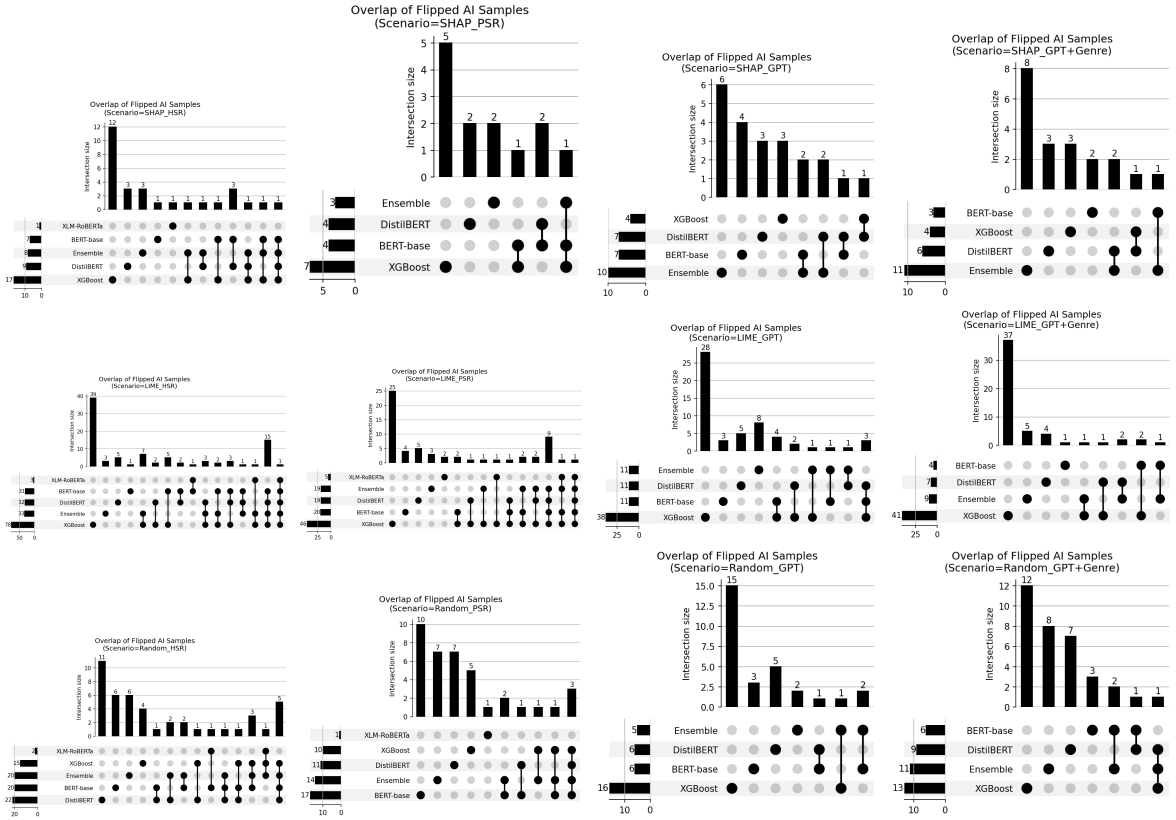


Fig. 2. Overlaps among flipped AI samples under different explainability methods and rewriting strategies. Each bar corresponds to the number of samples simultaneously flipped by one or more models.

In general, GPT-based replacements make text more natural or domain-specific, which greatly impacts models that rely on simple token patterns for detection. HSR and PSR (especially with SHAP/LIME) can also be effective while keeping a higher BLEU score, meaning fewer changes. Since PSR aligns part-of-speech tags, the text remains fluent but loses patterns AI might use, leading to performance drops in classifiers like XLM-RoBERTa and XGBoost. In contrast, our Ensemble model is more robust, as its combined transformer knowledge helps it detect AI content even after targeted token edits. Overall, our results show that the Ensemble model is the most accurate and resilient, effectively handling English and Dutch texts across different genres. While rewriting reduces detection rates, it does not fully bypass the Ensemble model.

## 5.1 Analysis of Ensemble Components

To understand the contribution of individual components to our ensemble's performance, we analyze the results from Table 4, which shows each single model's performance alongside the Ensemble. This analysis serves as a proxy for a leave one component out analysis, revealing how much each component contributes to the combined system.

*Ensemble vs. Single Models.* Across all language-domain combinations, the Ensemble consistently outperforms or matches the best single model. In English, the Ensemble achieves an average F1 of 0.86 compared to 0.78 for BERT-base, 0.84 for DistilBERT, and 0.77 for XLM-RoBERTa. In Dutch, the performance gap is even more pronounced: the Ensemble averages 0.78 F1 while BERT-base achieves only 0.55, DistilBERT 0.65, and XLM-RoBERTa 0.63. This demonstrates that the ensemble architecture provides substantial gains, particularly for challenging cross-lingual scenarios.

*Complementary Strengths.* Our ensemble benefits from the complementary capabilities of its constituent models. XLM-RoBERTa provides strong multilingual representations, making it valuable for Dutch text where monolingual BERT models struggle. BERT-base excels in English news and reviews where domain-specific patterns are important. DistilBERT offers computational efficiency while maintaining competitive performance. The frozen branch (pre-trained on AuTexTification) captures general AI-detection patterns, while the fresh branch (fine-tuned on augmented CLIN33) adapts to domain-specific characteristics. This dual-branch architecture enables the ensemble to leverage both broad knowledge transfer and targeted domain adaptation.

*Robustness Under Adversarial Conditions.* Table 5 reveals that the Ensemble exhibits smaller performance drops under token replacement attacks compared to single models. While XGBoost shows drops of up to 56% F1 under LIME+HSR, the Ensemble's maximum drop is 19%, demonstrating enhanced robustness. This resilience stems from the ensemble's redundant detection pathways—when one component is fooled by specific token manipulations, others may still correctly identify the AI-generated nature of the text.

*Limitations.* A complete leave one component out analysis would require retraining the ensemble with each component removed, which was beyond the scope of our current experimental setup. Future work could systematically evaluate the marginal contribution of each model by training multiple ensemble variants. Nevertheless, the single-model results provide strong evidence that the ensemble architecture meaningfully combines component strengths rather than merely averaging their predictions.

## 5.2 Sensitivity Analysis

Our token replacement framework raises questions about the relationship between the number of tokens modified and the resulting detection evasion. While a systematic parameter sweep across different token counts was not conducted in this study, we can draw insights from the observed patterns.

*Fidelity-Evasion Trade-off.* The similarity metrics in Table 6 reveal a clear trade-off between textual fidelity and evasion effectiveness. PSR-based methods achieve the highest BLEU scores (0.97 for SHAP+PSR), indicating minimal text modification, while producing moderate flip rates. In contrast, GPT-based methods show lower BLEU scores (0.76–0.83), reflecting more substantial rewrites, but do not necessarily achieve higher flip rates against the Ensemble. This suggests diminishing returns: beyond a certain modification threshold, additional changes may not improve evasion and could potentially introduce new AI-detectable patterns.

*Influence of Token Selection Strategy.* Comparing SHAP, LIME, and Random token selection reveals that targeting influential tokens (as identified by XAI methods) produces larger detection drops with fewer modifications. Under the same replacement strategy, LIME-selected tokens generally cause greater performance degradation than random selection, confirming that explainability-guided attacks are more efficient than random perturbations. The flip rates in Table 7 show that LIME+HSR achieves higher evasion against XGBoost with similar textual changes compared to Random+HSR.

*Token Count Considerations.* Our experiments replaced the top-K most influential tokens as identified by SHAP or LIME, where K was determined by the importance threshold rather than a fixed count. Based on prior work [48], we expect that most detection signal concentrates in the top 5–15% of tokens. Future work could systematically

vary K to characterize the sensitivity curve and identify optimal modification levels that balance evasion success against textual degradation.

### 5.3 Human Evaluation

To complement automated metrics, we conducted a small-scale human evaluation to assess the perceptual quality of rewritten texts and human ability to detect AI-generated content.

*Setup.* We selected 100 text samples (approximately 10% of the experimental corpus) stratified by rewriting strategy and model flip patterns: samples from each major rewriting condition (HSR, PSR, GPT, GPT+Genre), plus original AI texts and human-written controls from the same domains. The distribution included 58 samples that successfully evaded detection ("flipped": 18 HSR, 14 PSR, 14 GPT, 12 GPT+Genre), 12 that remained detected despite rewriting, 15 unmodified AI-generated texts, and 15 human-written texts. Samples were selected to represent diverse scenarios from our overlap analysis, including cases where multiple models were fooled and cases where only specific models were affected.

Two evaluators, blind to the experimental condition, rated each text on three dimensions: (1) *Fluency* (1–5 scale: how natural does the text read?), (2) *Coherence* (1–5 scale: is the text logically consistent?), and (3) *AI Detection* (5-point scale from "Definitely Human" to "Definitely AI"). We report inter-rater reliability using Cohen's $\kappa$.

*Results.* Table 8 summarizes the human evaluation findings.

Table 8. Human Evaluation Results (N=100 samples, 2 evaluators). Fluency and Coherence are rated on 1–5 scales. Detection shows the percentage of correct AI/human classifications by human evaluators. Cohen's $\kappa$ for inter-rater reliability: Fluency=0.12, Coherence=0.14, Detection=0.08.

| **Condition** | **N** | **Fluency** (1-5) | **Coherence** (1-5) | **Detection** (% Correct) |
|---|---|---|---|---|
| Original AI | 15 | 4.18 (±0.76) | 4.25 (±0.52) | 58.3% |
| Human Control | 15 | 4.47 (±0.58) | 4.65 (±0.51) | 88.3% |
| Rewritten (Flipped) | 58 | 3.95 (±0.84) | 4.02 (±0.89) | 34.5% |
| Rewritten (Detected) | 12 | 3.85 (±0.62) | 3.58 (±0.74) | 58.3% |
| *Overall* | 100 | 4.02 | 4.08 | 47.5% |

*Observations.* The human evaluation reveals several key findings. First, human evaluators achieved only 47.5% overall detection accuracy, substantially lower than the Ensemble model's 83% accuracy on the same samples, suggesting that machine detectors currently outperform average human readers at identifying AI-generated text. Second, texts that successfully evaded machine detection ("Flipped" condition) also fooled humans at high rates, with only 34.5% correctly identified as AI-generated—compared to 58.3% for original AI texts. This correlation between machine and human vulnerability suggests that our token replacement strategies effectively humanize AI text along dimensions that both humans and machines rely on for detection. Third, all conditions maintained high fluency (3.85–4.47) and coherence (3.58–4.65) ratings, indicating that rewriting strategies preserve text quality while reducing detectability. The modest inter-rater reliability ($\kappa \approx 0.11$) reflects the inherent difficulty of AI detection for non-expert readers, consistent with findings from prior human evaluation studies in AI-generated text detection [6].

## 6 Discussion and Conclusion

Our findings show that explanation techniques such as SHAP and LIME can guide systematic token replacements that make AIGT less likely to be classified as *AI-generated*. When we identify and edit tokens most indicative

of artificial content, simpler models like XGBoost can be misled at high rates, especially under LIME-based rewrites combined with synonym substitutions. Yet even more advanced models, such as BERT-base or DistilBERT, still show vulnerability in certain scenarios, particularly when GPT-generated replacements transform the text more extensively. These methods, however, do not completely deceive our Ensemble model, which remains comparatively robust across languages, genres, and rewriting methods.

*On the Significance of Performance Differences.* While the absolute performance differences between the Ensemble and individual models may appear modest in isolation, several factors underscore their practical importance. First, our statistical analysis confirms that these differences are statistically significant (Section 4.1), with the Ensemble consistently outperforming baselines across diverse experimental conditions. Second, when deployed at scale—processing thousands or millions of texts daily in real-world content moderation scenarios—even small percentage improvements translate to substantial reductions in false positives and negatives. Third, and perhaps most importantly, the Ensemble's primary advantage lies not in raw accuracy but in its *consistency*: it exhibits lower variance across languages, domains, and adversarial strategies than any single model. This robustness property is crucial for practical deployment, where unpredictable failure modes pose greater risks than marginally lower average performance.

*Contributions and Novelty.* This work makes several distinct contributions to the growing field of AI-generated text detection. To our knowledge, this is the first systematic study to leverage explainability methods for both adversarial evasion *and* robust detection within a unified framework. While prior work has explored either XAI-guided attacks [8] or detection systems in isolation, our dual perspective provides unique insights into the arms race between detection and evasion. Additionally, our frozen-plus-fresh ensemble architecture represents a novel design that balances the stability of pre-trained representations with domain-specific adaptability. Our cross-lingual (English and Dutch), multi-domain (news, reviews, social media) evaluation further distinguishes this work from studies limited to English-only or single-domain settings. Finally, the detailed analysis of token replacement strategies—from conservative synonym substitution to aggressive LLM-based rewriting—offers actionable guidance for both detection system designers and policymakers concerned with AI content transparency.

From the perspective of text similarity, we see that methods like HSR and PSR typically achieve higher BLEU and ROUGE scores, indicating minimal alterations to the original text. This can preserve meaning and style, but targeted token swaps can still hide enough AI signals to disrupt certain detectors. In contrast, GPT-based rewrites, especially GPT+Genre, often provide more comprehensive revisions, making the text harder to classify but also changing more from the source. Although such extensive rewriting can produce higher flip rates, our Ensemble classifier still maintains better overall performance than single-model systems.

In practice, these results underscore both the potential and the limitations of adversarial rewriting. Minor edits guided by SHAP or LIME can undermine certain classifiers. Future work might extend to more sophisticated paraphrasing of entire sentences or incorporate style transfer models that better replicate a specific human author. At the same time, it is crucial to consider watermarking and policy frameworks that ensure transparency when AIGT is involved. Our experiments show that even when AI content is skillfully masked, detection remains possible with multi-architecture approaches.

*Limitations and Future Directions.* Several limitations of this study merit acknowledgment. First, the CLIN33 dataset, while carefully curated, is relatively modest in size; larger-scale evaluations would strengthen confidence in our findings. Second, the AI-generated content in our datasets was produced by GPT-3.5 and BLOOM-series models, which represent an earlier generation of LLMs. The detection landscape is evolving rapidly, and content from newer models such as GPT-4, Claude, Llama 3, and their variants may exhibit different patterns that could affect both detection accuracy and adversarial vulnerability. Third, our evaluation covers two languages (English and Dutch) and three domains; generalization to other languages, particularly those with different morphological

structures, and to specialized domains such as scientific writing or legal text remains to be demonstrated. Fourth, while we provide computational cost estimates, we did not conduct a systematic study of the efficiency-accuracy trade-off under deployment constraints. Future work should address these gaps by: (1) evaluating on larger, more diverse datasets including content from the latest LLMs; (2) exploring additional languages and domains; (3) investigating combinations of XAI-guided rewriting with other adversarial techniques; (4) developing adaptive detection systems that can co-evolve with increasingly sophisticated evasion methods; and (5) extending human evaluation with larger, more diverse annotator pools including domain experts and professional content moderators, which would provide deeper insights into the aspects that distinguish AI-generated from human-written text.

In conclusion, our study shows how explanation methods can inform token replacements to disguise AIGT, yet also reveals that robust ensemble classifiers are harder to deceive. Balancing improved detection methods with the need for responsible use of writing aids will remain an important challenge. We hope these findings contribute to a deeper understanding of both the potential vulnerabilities in existing AI detection models and the methods that can keep them resilient against adversarial transformations.

## Acknowledgments

We thank the CLIN33 shared task organizers for making the dataset available, and we appreciate the helpful feedback from colleagues who guided our research directions. We also thank SURF for providing the computational facilities. We are grateful to Tina Shahedi for her contribution to the human evaluation study.

## References

[1] Kenza Amara, Rita Sevastjanova, and Mennatallah El-Assady. 2024. SyntaxShap: Syntax-aware Explainability Method for Text Generation. In *Findings of the Association for Computational Linguistics: ACL 2024*. 7336–7352.

[2] Guangsheng Bao, Yanbin Zhao, Zhiyang Teng, Linyi Yang, and Yue Zhang. 2023. Fast-detectgpt: Efficient zero-shot detection of machine-generated text via conditional probability curvature. *arXiv preprint arXiv:2310.05130* (2023).

[3] Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150* (2020).

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, et al. 2020. Language Models are Few-Shot Learners. *ArXiv* abs/2005.14165 (2020). https://api.semanticscholar.org/CorpusID:218971783

[5] Miranda Christ, Sam Gunn, and Or Zamir. 2024. Undetectable watermarks for language models. In *The Thirty Seventh Annual Conference on Learning Theory*. PMLR, 1125–1139.

[6] Elizabeth Clark, Tal August, Sofia Serber, Nikita Haduong, Suchin Xu, and Noah A Smith. 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 (Volume 1: Long Papers)*. 7282–7296.

[7] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555* (2020).

[8] Isaac David and Arthur Gervais. 2025. AuthorMist: Evading AI Text Detectors with Reinforcement Learning. *arXiv preprint arXiv:2503.08716* (2025).

[9] Pieter Fivez, Walter Daelemans, Tim Van de Cruys, , et al. 2024. The CLIN33 Shared Task on the Detection of Text Generated by Large Language Models. *Computational Linguistics in the Netherlands Journal* 13 (2024), 233–259.

[10] Kathleen C Fraser, Hillary Dawkins, and Svetlana Kiritchenko. 2025. Detecting ai-generated text: Factors influencing detectability with current methods. *Journal of Artificial Intelligence Research* 82 (2025), 2233–2278.

[11] Leon Fröhling and Arkaitz Zubiaga. 2021. Feature-based detection of automated language models: tackling GPT-2, GPT-3 and Grover. *PeerJ Computer Science* 7 (2021), e443.

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

[13] Xun Guo, Yongxin He, Shan Zhang, Ting Zhang, Wanquan Feng, Haibin Huang, and Chongyang Ma. 2024. Detective: Detecting ai-generated text via multi-level contrastive learning. *Advances in Neural Information Processing Systems* 37 (2024), 88320–88347.

[14] Xiaomeng Hu, Pin-Yu Chen, and Tsung-Yi Ho. 2023. RADAR: Robust AI-Text Detection via Adversarial Learning. *ArXiv* abs/2307.03838 (2023). https://api.semanticscholar.org/CorpusID:259501842

[15] Guanhua Huang, Yuchen Zhang, Zhe Li, Yongjian You, Mingze Wang, and Zhouwang Yang. 2024. Are AI-Generated Text Detectors Robust to Adversarial Perturbations?. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 6005–6024.

[16] John Kirchenbauer, Jonas Geiping, Yuxin Wen, et al. 2023. On the Reliability of Watermarks for Large Language Models. *ArXiv* abs/2306.04634 (2023). https://api.semanticscholar.org/CorpusID:259095643

[17] John Kirchenbauer, Jonas Geiping, Yuxin Wen, et al. 2023. A Watermark for Large Language Models. In *International Conference on Machine Learning*. https://api.semanticscholar.org/CorpusID:256194179

[18] Ryuto Koike, Masahiro Kaneko, and Naoaki Okazaki. 2024. Outfox: Llm-generated essay detection through in-context learning with adversarially generated examples. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 21258–21266.

[19] Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. 2023. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense. *Advances in Neural Information Processing Systems* 36 (2023), 27469–27500.

[20] Tharindu Kumarage, Amrita Bhattacharjee, Djordje Padejski, Kristy Roschke, Dan Gillmor, Scott Ruston, Huan Liu, and Joshua Garland. 2023. J-Guard: Journalism Guided Adversarially Robust Detection of AI-generated News. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*. 484–497.

[21] Guanlin Li, Yifei Chen, Jie Zhang, Jiwei Li, Shangwei Guo, and Tianwei Zhang. 2023. Towards the Vulnerability of Watermarking Artificial Intelligence Generated Content. *arXiv preprint arXiv:2310.07726* (2023).

[22] Weixin Liang, Mert Yuksekgonul, Yining Mao, Eric Wu, and James Zou. 2023. GPT detectors are biased against non-native English writers. *Patterns* 4, 7 (2023), 100779.

[23] Xiaoming Liu, Zhaohan Zhang, Yichen Wang, Hang Pu, Yu Lan, and Chao Shen. 2023. Coco: Coherence-enhanced machine-generated text detection under low resource with contrastive learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 16167–16188.

[24] Scott M. Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. In *Neural Information Processing Systems*. https://api.semanticscholar.org/CorpusID:21889700

[25] Eric Mitchell, Yoonho Lee, Alexander Khazatsky, et al. 2023. DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature. In *International Conference on Machine Learning*. https://api.semanticscholar.org/CorpusID:256274849

[26] Sandra Mitrovi'c, Davide Andreoletti, and Omran Ayoub. 2023. ChatGPT or Human? Detect and Explain. Explaining Decisions of Machine Learning Model for Detecting Short ChatGPT-generated Text. *ArXiv* abs/2301.13852 (2023). https://api.semanticscholar.org/CorpusID:256416337

[27] Hadi Mohammadi, Ayoub Bagheri, Anastasia Giahanou, et al. 2025. Explainability in Practice: A Survey of Explainable NLP Across Various Domains. *ArXiv* abs/2502.00837 (2025). https://api.semanticscholar.org/CorpusID:276094504

[28] Hadi Mohammadi, Anastasia Giachanou, and Ayoub Bagheri. 2024. A Transparent Pipeline for Identifying Sexism in Social Media: Combining Explainability with Model Prediction. *Applied Sciences* (2024). https://api.semanticscholar.org/CorpusID:272894586

[29] Hadi Mohammadi, Anastasia Giahanou, and Ayoub Bagheri. 2023. Towards Robust Online Sexism Detection: A Multi-Model Approach with BERT, XLM-RoBERTa, and DistilBERT for EXIST 2023 Tasks. In *Conference and Labs of the Evaluation Forum*. https://api.semanticscholar.org/CorpusID:264441492

[30] Harsha Moraliyage, Geeth Kulawardana, Daswin De Silva, and Damminda Alahakoon. 2025. Explainable Artificial Intelligence with Integrated Gradients for Detection of Adversarial Attacks on Text Classifiers. *Applied System Innovation* 8, 1 (2025), 17.

[31] L. Nacke et al. 2024. AI-authored abstracts perceived as more authentic than human-written ones. *Computers in Human Behavior: Artificial Humans* (2024).

[32] Ayat A Najjar, Huthaifa I Ashqar, Omar A Darwish, and Eman Hammad. 2025. Detecting AI-Generated Text in Educational Content: Leveraging Machine Learning and Explainable AI for Academic Integrity. *arXiv preprint arXiv:2501.03203* (2025).

[33] Duke Nguyen, Khaing Myat Noe Naing, and Aditya Joshi. 2023. Stacking the Odds: Transformer-Based Ensemble for AI-Generated Text Detection. *Language Technology Association* (2023), 173.

[34] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2016). https://api.semanticscholar.org/CorpusID:13029170

[35] G. Sadasivan et al. 2023. Can AI Detectors Detect ChatGPT? Analysis and Challenges. arXiv preprint `arXiv:2303.11156`.

[36] Areg Mikael Sarvazyan, José Ángel González, Marc Franco-Salvador, et al. 2023. Overview of AuTexTification at IberLEF 2023: Detection and Attribution of Machine-Generated Text in Multiple Domains. *Proces. del Leng. Natural* 71 (2023), 275–288. https://api.semanticscholar.org/CorpusID:262055483

[37] Steven Siddals, John Torous, and Astrid Coxon. 2024. "It happened to be the perfect thing": experiences of generative AI chatbots for mental health. *npj Mental Health Research* 3, 1 (2024), 48.

[38] Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, et al. 2019. Release strategies and the social impacts of language models. *arXiv preprint arXiv:1908.09203* (2019).

[39] Inhwa Song, Sachin R Pendse, Neha Kumar, and Munmun De Choudhury. 2024. The typing cure: Experiences with large language model chatbots for mental health support. *arXiv preprint arXiv:2401.14362* (2024).
[40] Jinyan Su, Terry Yue Zhuo, Di Wang, and Preslav Nakov. 2023. Detectllm: Leveraging log rank information for zero-shot detection of machine-generated text. *arXiv preprint arXiv:2306.05540* (2023).
[41] Vivek Verma, Eve Fleisig, Nicholas Tomlin, and Dan Klein. 2023. Ghostbuster: Detecting text ghostwritten by large language models. *arXiv preprint arXiv:2305.15047* (2023).
[42] Xi Yang, Kejiang Chen, Weiming Zhang, Chang Liu, Yuang Qi, Jie Zhang, Han Fang, and Nenghai Yu. 2023. Watermarking text generated by black-box language models. *arXiv preprint arXiv:2305.08883* (2023).
[43] Xianjun Yang, Wei Cheng, Yue Wu, Linda Petzold, William Yang Wang, and Haifeng Chen. 2023. Dna-gpt: Divergent n-gram analysis for training-free detection of gpt-generated text. *arXiv preprint arXiv:2305.17359* (2023).
[44] Catherine Yeh, Gonzalo A. Ramos, Rachel Ng, et al. 2024. GhostWriter: Augmenting Collaborative Human-AI Writing Experiences Through Personalization and Agency. *ArXiv* abs/2402.08855 (2024). https://api.semanticscholar.org/CorpusID:267658049
[45] Xiao Yu, Yuang Qi, Kejiang Chen, Guoqiang Chen, Xi Yang, Pengyuan Zhu, Weiming Zhang, and Nenghai Yu. 2023. Gpt paternity test: Gpt generated text detection with gpt genetic inheritance. *CoRR* (2023).
[46] Krystian Zawistowski. 2024. Unused information in token probability distribution of generative LLM: improving LLM reading comprehension through calculation of expected values. *arXiv preprint arXiv:2406.10267* (2024).
[47] Rowan Zellers, Ari Holtzman, Hannah Rashkin, et al. 2019. Defending Against Neural Fake News. *ArXiv* abs/1905.12616 (2019). https://api.semanticscholar.org/CorpusID:168169824
[48] Ying Zhou, Ben He, and Le Sun. 2024. Humanizing machine-generated content: evading AI-text detection through adversarial attack. *arXiv preprint arXiv:2404.01907* (2024).

## Reproducibility Checklist for JAIR

**All articles:**

(1) **All claims investigated in this work are clearly stated.**
*Answer:* Yes, we clearly state our main claims: we want to see how effective rewriting with SHAP/LIME is, and how an ensemble model performs.
(2) **Clear explanations are given how the work reported substantiates the claims.**
*Answer:* Yes, we explain how each experiment's results support our claims.
(3) **Limitations or technical assumptions are stated clearly and explicitly.**
*Answer:* Some. We focus on detection and rewriting only, not on other possible aspects.
(4) **Conceptual outlines and/or pseudo-code descriptions of the AI methods introduced in this work are provided, and important implementation details are discussed.**
*Answer:* Yes, our method flow is clearly described, especially for the ensemble approach in the algorithm 1.
(5) **Motivation is provided for all design choices, including algorithms, implementation choices, parameters, data sets, and experimental protocols beyond metrics.**
*Answer:* Yes, we explain why we use these models, datasets, and rewriting strategies.

## Articles containing theoretical contributions

*Does this paper make theoretical contributions?*
*Answer:* No. (We do not do formal theoretical proofs.)

## Articles reporting on computational experiments

*Does this paper include computational experiments?*
*Answer:* Yes.

(1) **All source code required for experiments is included or will be made publicly available upon publication, following best practices.**
*Answer:* We plan to share the main code base (e.g., on GitHub).
(2) **The source code comes with a license that allows free usage for reproducibility purposes.**
*Answer:* Yes, we use a permissive license.
(3) **The source code comes with a license that allows free usage for research purposes in general.**
*Answer:* Yes.
(4) **Raw, unaggregated data from all experiments is included or will be made publicly available.**
*Answer:* We will share our final splits, but some augmented data might be partial.
(5) **The unaggregated data comes with a license that allows free usage for reproducibility purposes.**
*Answer:* Yes, as far as allowed by the original dataset licenses.
(6) **The unaggregated data comes with a license that allows free usage for research purposes in general.**
*Answer:* Yes, within the limits of the original dataset licenses.
(7) **If an algorithm depends on randomness, the method used for generating random numbers and setting seeds is described sufficiently.**
*Answer:* We set random seeds but do not deeply document random generation details.
(8) **The execution environment for experiments (hardware/software) is described.**
*Answer:* Yes, we mention key libraries (e.g., Hugging Face Transformers, PyTorch).
(9) **The evaluation metrics used are clearly explained and motivated.**
*Answer:* Yes, we use F1, accuracy, BLEU, and ROUGE and explain why.

(10) **The number of algorithm runs used to compute each result is reported.**
*Answer:* We use cross-validation or mention repeated runs (varies).
(11) **Reported results have not been "cherry-picked."**
*Answer:* Yes, we show both successes and failures.
(12) **Analysis of results goes beyond single-dimensional summaries of performance.**
*Answer:* We show F1, accuracy, and text-similarity. (We do not always show standard deviation or confidence intervals.)
(13) **All (hyper-) parameter settings for the algorithms/methods used are reported, along with the rationale or method for determining them.**
*Answer:* Yes, we provide learning rates, batch sizes, etc.
(14) **The number and range of (hyper-) parameter settings explored prior to final experiments have been indicated.**
*Answer:* We mention random search, though not each detail.
(15) **Appropriately chosen statistical hypothesis tests are used to establish significance in the presence of noise.**
*Answer:* We do not do formal significance testing.

### Articles using data sets

*Does this work rely on one or more data sets?*
*Answer:* Yes.

(1) **All newly introduced data sets are included in an online appendix or will be made publicly available with a suitable license.**
*Answer:* We mostly use public sets (CLIN33, AuTexTification) and will release our augmented version.
(2) **The newly introduced data set comes with a license that allows free usage for reproducibility purposes.**
*Answer:* Yes, for the augmented subset.
(3) **The newly introduced data set comes with a license that allows free usage for research purposes in general.**
*Answer:* Yes.
(4) **All data sets drawn from the literature or other public sources are accompanied by appropriate citations.**
*Answer:* Yes, we cite the original authors.
(5) **All data sets drawn from the existing literature are publicly available.**
*Answer:* Yes, with possible constraints.
(6) **All new or non-public data sets are described in detail, including relevant statistics, collection, and annotation processes.**
*Answer:* Yes, for our augmented data.
(7) **All methods used for preprocessing, augmenting, batching, or splitting data sets are described in detail.**
*Answer:* Yes, we detail how we clean and split.

### Explanations on any of the answers above (optional):

We use mostly public data and provide augmented sets where possible. Some data licensing is inherited from the original corpora. Where possible, we provide reproducible code, scripts, and subsets to ensure others can replicate our results.