Detecting Synthetic Lyrics with Few-Shot Inference

Anonymous ACL submission

Abstract

In recent years, generated content in music has gained significant popularity, with large language models being effectively utilized to produce human-like lyrics in various styles, themes, and linguistic structures. This technological advancement supports artists in their creative processes but also raises issues of authorship infringement, consumer satisfaction and content spamming. To address these challenges, methods for detecting generated lyrics are necessary. However, existing works have not yet focused on this specific modality or on creative text in general regarding machine-generated content detection methods and datasets. In response, we have curated the first dataset of high-quality synthetic lyrics and conducted a comprehensive quantitative evaluation of various few-shot content detection approaches, testing their generalization capabilities and complementing this with a human evaluation. Our best few-shot detector, based on LLM2Vec, surpasses stylistic and statistical methods, which are shown competitive in other domains at distinguishing human-written from machine-generated content. It also shows good generalization capabilities to new artists and models, and effectively detects post-generation paraphrasing. This study emphasizes the need for further research on creative content detection, particularly in terms of generalization and scalability with larger song catalogs. All datasets, pre-processing scripts, and code are available publicly on GitHub and Hugging Face under the Apache 2.0 license.

1 Introduction

In recent years, generated content has become increasingly widespread across various modalities, including audio (Kong et al., 2020), image (Ho et al., 2020), video (Singer et al., 2023; Thambiraja et al., 2023), and text (Brown et al., 2020). This technological progress has been influencing numerous fields, such as literature, visual arts, and entertainment. Music, in particular, has experienced a notable impact from this trend, with the emergence of tools like Suno AI[1] and ChatGPT (OpenAI, 2023) facilitating faster and more accessible content creation by generating lyrics (Nikolov et al., 2020; Qian et al., 2023; Tian et al., 2023) and audio (Copet et al., 2023), thereby broadening the scope of artistic activities and creation.

The widespread adoption of accessible Large Language Models (LLMs) like BLOOM (Scao et al., 2023), Mistral (Jiang et al., 2023), ChatGPT (OpenAI, 2023) and LLaMa 2 (Touvron et al., 2023) has the potential to transform the way creative text is written. These freely available models can produce text with human-like quality at minimal cost, making them highly accessible for creative tasks such as writing poems (Popescu-Belis et al., 2023), song lyrics (Qian et al., 2023), movie scripts (Zhu et al., 2023b), and other types of creative content (Swanson et al., 2021; Chakrabarty et al., 2024). The advent of machine-generated text offers a wealth of possibilities for artists, providing new sources of inspiration and a means to overcome creative blocks (Zhu et al., 2024; Behrooz et al., 2024). In the music domain, by leveraging these advanced LLMs, songwriters could create content on diverse themes, styles and linguistic structures.

However, this widespread adoption of LLMs also raises concerns about authorship infringement, consumer satisfaction [2] and content spamming. To manage the dissemination of such content and prevent potential abuses, it is crucial to develop methods for detecting machine-generated lyrics. But yet, no related works focus on detecting machine-generated creative content like poems or lyrics, despite their significantly different nature from other types of documents. This difference stems from their unique semantic and rhythmic structure, as well as the multiple socio-cultural references they

[1] suno.com

[2] community.spotify.com/t5/Content-Questions/Release-Radar-this-week-was-almost-all-AI-generated-music/td-p/5630466

convey (Spanu, 2019).

Various approaches have been considered in the past to tackle this detection task, and are often modeling it as a binary classification problem (Mitchell et al., 2023), where detectors distinguish between human-written and machine-generated content from various black-box systems. However, none of them have been assessed on creative content and are often limited in terms of data and model generalization (Uchendu et al., 2020; Bakhtin et al., 2019).

In order to overcome this, we propose the following contributions:

- The curation and release of the first dataset of human-like synthetic lyrics, allowing to perform machine-generated content detection.
- A quantitative study of seven few-shot content detection approaches including LLM2Vec, UAR, Min-K % Prob and Shannon entropy on this new type of data, lyrics.
- An evaluation of the detectors regarding their capacity to generalize to newer artists and generative models of different sizes, trained with various procedures, and a study of the impact of the paraphrasing attack on the detectors.
- A human evaluation on the detection of machine-generating lyrics.

The datasets, pre-processing scripts, and codes related to this work are accessible on Hugging Face [3] and Github [4] under Apache 2.0 license in compliance with content copyrights.

2 Related Work

The detection of machine-generated content is a well-established task (Lavergne et al., 2008; Badaskar et al., 2008), with its origins in various fields of machine learning (Rana et al., 2022; Ahmed et al., 2022; Zhou and Lim, 2021; Guarnera et al., 2024; Bammey, 2024). Historically, the focus has been on identifying generated content across different modalities such as newspapers and scientific articles for text, or voice spoofing for audio. However, recent advances in generative model quality and creativity have highlighted the need for detectors capable of handling more sophisticated forms of text, such as creative content. The music industry, in particular, faces multiple modalities vulnerable to machine-generated content, with current efforts primarily addressing audio detection (Zang et al., 2024; Wu et al., 2017; Afchar et al., 2024). This has led to a significant gap in the literature regarding the detection of machine-generated textual creative content such as lyrics, compelling us to focus first on general methods for detecting machine-generated text.

Machine-generated text detection is commonly formulated as a classification task (Liu et al., 2023; Huang et al., 2024). One way of solving it is to use supervised learning, where classification models based on textual encoders (Abburi et al., 2023; Wu et al., 2023; Pu et al., 2023; Wang et al., 2023) or LLMs (Macko et al., 2023; Antoun et al., 2024; Chen et al., 2023; Kumarage et al., 2023) are trained on a dataset containing both machine-generated and human-written texts. However, those supervised models are trained to explicitly detect a very particular set of machine-generated data and may suffer from over-fitting issues on unseen data (Uchendu et al., 2020; Bakhtin et al., 2019), such as newer artists or generative models.

In a different line of research, attempts have been made to differentiate between machine-generated and human-written texts based on statistical anomalies in the entropy or perplexity by estimating token-level log probabilities (Su et al., 2023; Zhu et al., 2023a; Sadasivan et al., 2024). Other approaches, like DetectGPT (Mitchell et al., 2023), found that machine-generated texts can be detected after a few intensive perturbations, outperforming previous methods. Parallel studies explored watermark-based detection (Abdelnabi and Fritz, 2021; Chakraborty et al., 2023; Kirchenbauer et al., 2023), but these methods suffer from the need to access model logits, which is not feasible for models available exclusively through APIs, such as GPT-4 Turbo or Claude 3.

3 Datasets

As highlighted in related works, most of the text detection studies have focused on textual data of a very different nature than lyrics in terms of structure, semantics and vocabulary. Consequently, there is currently a lack of validated corpus suitable for detecting machine-generated lyrics. Moreover, most studies on machine-generated content detection use generated data by simply relying on LLMs outputs, without validating the content. These studies often do not evaluate the soundness of generated data by humans nor explicitly mention post-generation steps where generation artifacts or sparse character encodings could be removed from

[3] Hidden for double-blind.
[4] Hidden for double-blind.

the text. For our experiments, we require a dataset comprising two types of lyrics: human-written and machine-generated. Due to the absence of such a dataset, we opted to construct one ourselves.

Our goal with this dataset is to develop methods based on a single generative model and three artists, to assess the scalability of our approaches across six additional models and two more artists. To do so, we started curating human-written lyrics from a music metadata provider, LyricsFind [5]. The machine-generated lyrics were automatically generated by seven large language models (LLMs) under controlled conditions, as we will explain in further sections. We chose this approach to focus exclusively on text generation, avoiding the use of other types of lyric generators that incorporate multiple modalities such as melody or audio (Qian et al., 2023; Tian et al., 2023).

3.1 Human-written Lyrics Collection

To narrow the scope of our study, we first focus on five well-known English-speaking artists: Drake, Ed Sheeran, Post Malone, Taylor Swift, and The Weeknd. These artists were selected based on the previous year's Billboard "Top Artists" lists [6]. This selection allows us to obtain a substantial amount of lyrics from the provider.

We constrained the curated lyrics to those available on the provider's server and filtered them so that they must have been released within the past year and a half. This prevents data leakage into the models used for our detectors.

Additionally, we filtered the lyrics by language, ensuring that only English lyrics were included. So, we collected language tags given by the provider and used our language tagger, trained on 50 languages from MASSIVE (FitzGerald et al., 2023), to double-check the content's language, since some mistakes were present in the providers data. We also implemented a deduplication process to avoid multiple variations of the same song due to different delivery formats (album, single, live, etc.), based on the text.

3.2 Synthetic Lyrics Generation

The quality of the generated outputs influences the difficulty of the task and the system's generalization capability. To maintain high-quality and human-like lyrics, we employed a four-step process. We empirically evaluate each step's output by visually inspecting it for potential issues or generation artifacts, and we continuously enhance the normalization and filtering steps accordingly.

Step 1 - Generation. The first step consists of generating a few thousand lyrics using each of the 7 models from Section 3.4 by using the prompt listed in the Appendix A and conditioned during in-context learning with 3 lyrics from the same artist and taken from the 150 human-written lyrics (Table 1). To ensure the generated lyrics closely resemble the real ones, we instruct the model to adhere to the same structure and guidelines as outlined by the provider's formatting guidelines [7], listing these guidelines as bullet points in the prompt. Furthermore, to maximize the use of our available data, we condition the models by varying the order of the few-shot examples, as demonstrated in the work of Lu et al. (2022), which acts as a seed and diversifies the model's outputs. The hyperparameters used to generate the lyrics are listed in Table 6 and all the models are quantized in GGUF Q4 to ensure they can run efficiently on an 8GB M1 MacBook, suitable for an isolated individual using consumer-grade hardware with reasonable inference times.

Step 2 - Normalization. Next, we normalize the generated lyrics using various regular expressions developed along the process and based on the model's output, in order to prevent artifacts absent from real lyrics. For instance, we remove punctuation at the end of sentences when needed, eliminate quotations, and remove references to the generation process (such as "note: the lyrics generated," "written in the style," or "here's an example of a song") as well as indications of offensive content (e.g. "I cannot generate inappropriate").

Step 3 - Filtering. After, we sample the generated lyrics to align with the typical style of the artists by employing statistical metrics drawn from their real lyrics. It includes parameters like sentence length, number of verses, verse size, and overall word count. During generation, each metric distribution of the human-written lyrics is represented in box plots, and the generated content must fall within the interquartile range to be preserved.

Step 4 - Semantic similarity. Lastly, we conduct a semantic similarity comparison between the generated lyrics and the human-written ones, retaining up to 150 synthetic lyrics that are the most semantically similar to real ones for each artist / generative model combination. This semantic similarity is performed using Sentence Transformers (Reimers and

[5] lyricfind.com
[6] billboard.com/charts/year-end/top-artists
[7] docs.lyricfind.com/LyricFind_LyricFormattingGuidelines.pdf

Gurevych, 2019) library and `all-MiniLM-L6-v2` (Wang et al., 2021) model.

3.3 Data split

Human-written lyrics from three artists (Drake, Post Malone and Ed Sheeran) were used for the few-shot generation of synthetic lyrics (Section 3.2) and the few-shot detectors (Section 4). Specifically, for the detection methods, we subsample 300 lyrics equally distributed across these artists with 50 human-written and 50 machine-generated lyrics obtained using only LLaMa 2 13B for each of them. Relying on only one model during the detection is made to allow us to test the generalization capabilities of the detectors across newer models.

We reserved two artists (The Weeknd and Taylor Swift) exclusively for the evaluation to demonstrate the detector's generalization capabilities. The final evaluation set comprises 4,572 synthetic lyrics and 625 human-written ones, with the artist distribution as follows in Table 1:

	Artists	Generated	Human-written
	Vector Space ("Train")		
Seen (S)	*Drake*	$50^{\dagger}$	50
	Post Malone	$50^{\dagger}$	50
	Ed Sheeran	$50^{\dagger}$	50
	Evaluation ("Test")		
Seen (S)	*Drake*	931	128
	Post Malone	769	42
	Ed Sheeran	902	84
Unseen (U)	*Taylor Swift*	922	153
	The Weeknd	898	68
	Total	4,572	625

Table 1: Distribution of the dataset labels across artists. †LLaMa 2 13B is the only model seen as machine-generated lyrics in the vector space, while all the 7 models are available in the test set.

3.4 Generative Models

Various types of autoregressive LLMs have been used to generate the synthetic lyrics. Our choice was guided by the need for diversified architectures, model sizes and training procedures, in order to include content from various models runnable on consumer-grade hardware and ensure generalization. Three model types have been selected:

Foundation Models. The foundation models are LLMs trained from scratch using a wide range of data crawled from the web. LLaMa 2 13B (Touvron et al., 2023), LLaMa 3 8B (AI@Meta, 2024) and Mistral 7B (Jiang et al., 2023) are the three models we selected due to their very interesting performances to size ratio.

Small Language Models These models offer performances very close to those from previous foundation models while being much smaller. We only focused on TinyLLaMa 1.1B (Zhang et al., 2024) because other models of the same size such as Phi 1.5 (Li et al., 2023) or Pythia 1.4B (Biderman et al., 2023), could not consistently generate lyrics without any form of hallucination.

Instruction-tuned Lastly, the instruction-tuned models are models based on the weights of the previous foundation models and fine-tuned on synthetic instructions that look very similar to human prompts. We selected three models: First, WizardLM2 7B (Xu et al., 2024), which is based on Mistral 7B and fine-tuned using DPO (Rafailov et al., 2023) on 250K human-like instructions obtained from the 52K instructions of Alpaca (Taori et al., 2023). Vicuna 7B (Chiang et al., 2023) is based on LLaMa 2 7B and fine-tuned on 70K user-shared conversations collected from ShareGPT, a website where people share their ChatGPT interactions. Zephyr 7B (Tunstall et al., 2023), derived from Mistral 7B, has been fine-tuned on two datasets: UltraChat (Ding et al., 2023), which includes 1.47 million multi-turn dialogues, and UltraFeedback (Cui et al., 2024), which comprises 64,000 instructions.

4 Text Detection Methods

Detecting machine-generated lyrics can be approached as a binary classification task, categorizing them into two classes: "human-written" and "machine-generated" based on a set of features. One effective method for this is to fine-tune an encoder-based classifier like BERT on a set of reference data (Liu et al., 2023). This classifier can then provide a probability distribution for new lyrics that need to be classified. Another approach involves using the k-nearest neighbors (k-NN) algorithm. This method builds a dynamic vector index using lyrics referenced over time. When new lyrics need to be analyzed, a distance-based algorithm retrieves the k closest points to the query and typically returns the most frequent label among them.

The first approach, while powerful, requires retraining from scratch whenever new ground-truth data is available, and it has limitations in explainability and in controlling the impact of individual features. Conversely, the k-NN approach is more suited for low-resource environments, facilitating detection even with a limited number of lyrics (few-shot) and allowing for continuous system updates by incorporating new examples of machine-generated and human-written lyrics, thereby improving detection performance over time as lyrics are manually flagged by communities or editorial teams. Making us motivated to use it.

In the following sections, we introduce a wide range of features used to characterize the limited number of lyrics when projected into the vector space and utilized to make the few-shot k-nearest neighbor classification.

4.1 Random baseline

For each of the lyrics, we randomly attribute one of the two classes, "human-written" or "machine-generated".

4.2 Maximum Negative Log-Likelihood

This approach consists of computing the token's level negative log-likelihood of the lyrics by using an autoregressive LLM. In our case, we selected a model that is not in the list of models used to generate the lyrics, specifically `LLaMa 2 7B`.

Since lyrics are composed of individual verses which can be altered independently through human intervention (e.g. post-editing in the creative process or to fool a detector in case of an attack), we decided to compute the log probabilities of the tokens considering only the verse at the time as our context. It also allows us to distribute the computation and get faster inference time. Once we obtain those token's level negative log-likelihood, we take the maximum value across all the verses and use it as a one-dimensional vector of statistical features for the lyrics.

4.3 Perplexity

The perplexity (PPL) consists of getting a lyrics level metric obtained by an exponential average of the negative log-likelihood of the lyrics.

$$\mathrm{PPL}(X) = \exp\left\{-\frac{1}{t}\sum_{i}^{t}\log p_\theta(x_i|x_{<i})\right\}$$

Where $\log p_\theta(x_i|x_{<i})$ is the negative log-likelihood of the current token (x_i) considering the past tokens window ($x_{<i}$) of the sequence (t).

It allows us to encapsulate the overall probability of the lyrics being formulated as they are into a single value. The higher the perplexity (PPL), the less likely it is that the lyrics are human-written. However, there can be exceptions, as artistic writing may be more "surprising".

4.4 Shannon Entropy

The Shannon entropy is used as a measure of the self-information (Shannon, 1948) inside the probability distributions of a sequence. In the case of the lyrics and token's level negative log-likelihood, this information represents the sparsity or the diversity of the vocabulary used to construct the verse.

$$H(X) = -\sum_{x\in X} p(x)\log p(x)$$

We combine both the highest and lowest entropy computed per lyrics verse in a 2-dimensional vector, in order to capture the intrinsic variability of the lyrics and have a comprehensive overview of it.

4.5 Min-K% Prob

The Min-K% Prob method, introduced by Shi et al. (2024), involves selecting a subsample of K% of the lowest token-level negative log-likelihood probabilities from the entire set of lyrics. These probabilities are then averaged to create a single one-dimensional document-level feature. In our case, we selected a $K = 10$, based on the observations in Appendix F

4.6 Semantic and Syntactic Embeddings

We used dense semantic embeddings obtained using the library Sentence Transformers (SBERT) by Reimers and Gurevych (2019) and the model `all-MiniLM-L6-v2` from Wang et al. (2021) in order to capture the differences in the semantic and syntactic structure (Jawahar et al., 2019) of the lyrics written by humans and machines.

4.7 Authorship Embeddings

Unlike SBERT embeddings, the Universal Authorship Representation model (UAR) from Soto et al. (2021) generates stylistic representations of lyrics by capturing the author's writing styles. Soto et al. (2024) adapted this model to determine whether a given text was written by a human or not. For this purpose, UAR embeddings were fine-tuned using a contrastive learning approach, with positive samples from one Reddit user and negative samples from another. Two types of datasets were used, resulting in the MUD and CRUD model variants, which were trained on data from 1 and 5 million distinct Reddit users respectively.

4.8 LLM2Vec

LLM2Vec is an unsupervised method designed to convert auto-regressive LLMs into text encoders. This transformation is achieved through

Detectors	LLaMa 2 13B† (Lyrics Generators)		LLaMa 3 8B		Mistral 7B		TinyLLaMa		Vicuna		WizardLM2		Zephyr		Human-written		
Artists	S	U	S	U	S	U	S	U	S	U	S	U	S	U	S	U	Overall
Random	45.37	42.86	52.00	53.67	51.33	49.00	50.16	48.75	52.89	45.33	46.87	53.33	53.11	49.67	47.98	41.34	47.52
Statistics Methods																	
Perplexity	62.33	48.70	67.78	67.00	78.89	84.00	57.96	45.33	63.78	53.33	71.92	72.67	68.44	73.33	57.20	53.59	60.77
Max. Neg. Log-Likelihood	**89.17**	**89.61**	80.89	85.33	75.78	74.33	**77.58**	**72.33**	**90.44**	**90.00**	63.19	55.67	81.56	78.33	**83.44**	**89.38**	**82.44**
Shannon Entropy																	
Max	73.70	80.68	91.56	**95.00**	88.22	**94.00**	50.60	58.92	68.89	76.33	71.64	73.00	70.00	76.00	77.44	71.24	75.36
Max+Min	78.16	81.66	**94.44**	91.33	88.44	88.67	64.63	60.17	87.33	80.33	68.57	65.33	72.22	78.33	80.61	82.76	80.06
Min-K% Prob (k=10)	73.41	56.33	88.22	90.00	**92.44**	93.67	70.50	51.00	73.56	63.00	**93.19**	**96.67**	**88.00**	**90.67**	70.73	88.56	79.23
Embeddings Methods																	
SBERT	84.45	68.02	89.11	87.67	86.89	94.33	54.74	55.17	74.22	62.33	**87.88**	**91.67**	78.22	69.00	74.84	73.53	76.06
LLM2Vec																	
LLaMa 3 8B	**90.31**	86.53	**97.56**	**98.33**	**95.11**	**96.67**	**70.03**	**59.42**	**92.22**	**90.67**	78.32	80.00	**90.44**	87.67	94.73	95.59	**90.97**
LLaMa 2 7B	82.87	82.47	95.33	97.67	77.78	88.00	57.46	45.33	86.67	83.33	45.07	48.33	72.67	74.33	**97.63**	90.77	84.48
UAR																	
CRUD	69.18	**88.80**	68.67	64.33	74.67	81.00	32.84	32.92	47.56	48.67	44.81	44.67	63.33	81.67	90.64	89.13	74.83
MUD	79.89	85.23	65.56	43.67	84.22	88.00	32.69	37.42	46.44	40.67	53.19	59.00	72.67	**93.33**	95.39	**95.75**	79.18

Table 2: Recall of the 7 generator models (x-axis) against human-written lyrics by each of the 8 detection methods (y-axis). S refers to the artists seen in the vector space and U to the unseen ones. The overall micro recall score between human-written and machine-generated labels is reported in the last column. For each category of detectors, the best-performing one is in bold and the second-best is underlined. †LLaMa 2 13B is the only model seen in the vector space (Table 1).

a three-step process inspired by techniques from BERT (Devlin et al., 2019) and Sentence Transformers (Reimers and Gurevych, 2019). The steps include enabling bidirectional attention, implementing masked next-token prediction, and applying unsupervised contrastive learning. Together, these steps enhance the model's ability to capture meaningful textual information as a vector.

5 Experiments

The machine-generated lyrics detection can be modeled as a binary classification problem, similar to other types of generated text detection mentioned in the related works. The first class is "human-generated" which refers to the original lyrics extracted from the provider. The second class is "machine-generated" and refers to the synthetic lyrics automatically generated by the LLMs.

5.1 Metrics

To evaluate the performance of our detectors and their capacity to generalize to new generative models and artists, we choose to use the recall for each of each generator / detector combination and the micro-recall computed over both classes "human-written" and "machine-generated", thus mentioned as the "overall" metric in Table 2.

Recall is a more suitable metric than accuracy because we want to minimize false negatives for human-generated lyrics while maximizing true positives for synthetic lyrics and the overall metric. In the context of a synthetic content detection system, it is less problematic to confuse synthetic lyrics with human-written ones, than to incorrectly classify human-written lyrics as generated.

5.2 Results

First, we observe in Table 2 that no single model excels equally across all generators and artists, and none of the generators consistently outperforms the detectors. The performance difference during the evaluation between artists seen (S) in the k-NN and those unseen (U) as referred to in Section 1, depends on the generator and detector used. Unsurprisingly, artists not represented in the vector space tend overall to perform worse than those that are. For generators, there is no clear pattern related to their presence in the vector space, while the only seen generator, LLaMa 2 13B, performs similarly to unseen ones, which shows the generalization of our approaches to different generative models.

However, some generators, such as WizardLM2 and TinyLLaMa, are less frequently detected, possibly due to their different architectures capturing linguistic properties better, resulting in higher-quality lyrics. On the other hand, foundation models like LLaMa 2 13B, Mistral 7B or some instruction-tuned models like Vicuna and Zephyr are more frequently detected by both statistical and embeddings-based methods, indicating a bad generalization than other types of models which are aimed at human-like interactions such as WizardLM2.

We also observe significant differences among detectors in their ability to correctly label human-written lyrics, a critical criterion discussed in Section 5.1. Some detectors tend to mislabel human-written lyrics as machine-generated, possibly due to a collapsed vector space that lacks sufficient differentiation. Specifically, the Shannon entropy, LLM2Vec, and UAR-based approaches are particularly accurate and favor human-written labels, which makes them the most suitable detectors in our use case. Despite LLM2Vec embeddings built from LLaMa 2 7B being the most accurate for human-written lyrics, it is not the overall most effective embeddings-based method. It is worth noticing that LLaMa 3 8B outperforms LLaMa 2 7B by an overall difference of 6.49%. These LLM2Vec detectors significantly surpass others, including UAR embeddings, previously considered in the literature (Soto et al., 2024) as more effective compared to earlier methods like statistical approaches or SBERT. For UAR, MUD performs better than CRUD by 4.35%, highlighting the benefits of using embeddings built from a more diverse range of users. The best-performing statistical method is the maximum negative log-likelihood, and offers very good performances compared to UAR CRUD and SBERT. The same detector outperforms the Shannon entropy by 2.38% when using maximum and minimum features and the perplexity by 21.67%.

5.3 Human evaluation

The human evaluation aims to classify given lyrics as either "human-written" or "machine-generated" based solely on individual perception of the lyrics, fluency, coherence, and structure. We ensure that participants aren't familiar with the artists and haven't recognized many songs during the annotation process (Figure 4), in order to prevent having a biased experimental protocol. This approach allows for direct comparison with automatic detectors and addresses three questions:

- Is the quality of the generated lyrics sufficient to serve as ground truth?
- How do a small subset of humans perform on this task compared to the automatic detectors?
- Are there similarities in the criteria used by humans and detectors for their judgments?

To achieve this, a random subsample of 70 lyrics was selected, evenly split between generated and human-written, and uniformly distributed across different artists and models. Participants are unaware of this distribution to prevent bias and only see one isolated lyric at a time when annotating. They are asked to fill out a form where they have to select between one of the two classes: "human-written" or "machine-generated", and to rate their confidence on a scale from 1 to 4, as detailed in Appendix C. The performance of each participant and the detectors is detailed below in the Table 3:

Participant ID	Machine-generated		Human-written		Overall
	Recall	Confidence	Recall	Confidence	
Participant 1	54.28	3.14	97.14	3.68	75.71
Participant 2	40.00	2.20	43.44	2.05	41.72
Participant 3	57.14	2.20	78.50	2.42	67.82
Participant 4	74.28	2.37	82.88	2.45	78.58
Statistics Methods					
Perplexity	51.42	-	69.96	-	60.70
Negative Log-Likelihood	40.00	-	76.66	-	58.33
Shannon Entropy	60.00	-	70.04	-	65.02
Min-K% Prob (k=10)	54.28	-	50.71	-	52.50
Embeddings Methods					
SBERT	82.85	-	78.83	-	80.84
LLM2Vec					
LLaMa 2 7B	94.28	-	98.18	-	96.23
LLaMa 3 8B	97.14	-	91.36	-	94.25
UAR					
CRUD	65.71	-	100.00	-	82.86
MUD	68.57	-	94.84	-	81.71

Table 3: Human participants' performance on a subsample of 70 lyrics.

The standard deviation in participant's scores is substantial, with a difference of 14.53%. Participant 4 achieved the highest score at 78.58%, while Participant 2 had the lowest at 41.72%. This variability is likely due to Participant 2's difficulty in identifying distinguishable patterns to guide her decisions as mentioned in Appendix 4. Participants 1, 3 and 4 performed better than statistical methods but worse than embeddings-based methods. Overall, embeddings-based methods outperform humans, providing better performance on both classes simultaneously by capturing the sentence's inner structure and sequence sparsity, unlike humans who rely on superficial judgments.

Additionally, humans tend to identify human-written lyrics more accurately than machine-generated ones, a trend observed in 75% of the detectors. It is also evident that TinyLLaMa is the easiest generator for humans to identify, with a 90% recall, while LLaMa 3, WizardLM2, and Zephyr are the hardest, each with a 40% recall. These results indicate very different behaviors compared to those observed by detectors, reinforcing the idea that humans and machines assess lyrics very differently.

In terms of the agreement, participants completely agreed on the label 28.57% of the time. In the remaining 71.43% of cases, at least one participant disagreed with the others. This significantly reduced the values of Kappa Cohen and Gwet's AC1 as shown in Table 5, highlighting the task's difficulty and the divergences among participants. The Kappa scores involving Participant 2 indicate that annotations are very close to or worse than random, as negative Kappa and Gwet's AC1 val-

		Lyrics Generators																
	Generators Models	LLaMa 2 13B		LLaMa 3 8B		Mistral 7B		TinyLLaMa		Vicuna		WizardLM2		Zephyr		Human-written		
	Artists	*S*	*U*	*S*	*U*	*S*	*U*	*S*	*U*	*S*	*U*	*S*	*U*	*S*	*U*	*S*	*U*	Overall
	Statistics Methods																	
Detectors	*Max. Neg. Log-Likelihood*	90.45	85.55	83.78	85.33	79.11	76.33	78.76	73.83	89.78	89.67	73.37	63.67	83.56	81.33	83.44	89.38	**83.59**
	Shannon Entropy Min+Max	54.33	51.79	72.67	64.33	58.67	61.00	49.17	35.58	64.22	56.67	53.84	38.67	54.67	50.00	80.61	82.76	<u>68.43</u>
	Embeddings Methods																	
	LLM2Vec LLaMa 3 8B	91.80	94.16	98.00	97.67	93.11	99.67	78.48	76.08	91.33	94.67	79.45	96.33	85.56	94.33	94.73	95.59	**92.67**
	UAR - MUD	75.89	90.58	77.78	78.00	88.00	94.00	48.44	57.67	61.78	72.00	59.78	66.33	74.22	93.00	95.39	95.75	<u>84.35</u>

Table 4: Recall of the top two detectors from each category of approaches was assessed during the evaluation phase against paraphrasing attacks. To avoid overwhelming information, we presented results for only four models as they represent the overall trend observed. The highest value is highlighted in bold, while the second highest is underlined.

ues were calculated from the annotations, despite having well-defined criteria during annotation (Appendix H). We can also note that participants appeared to recognize a common Taylor Swift song, which slightly increased the agreement score (Figure 4).

When it comes to confidence scores, we can observe in Table 9 that participants tend to instinctively anticipate their mistakes by giving lower confidence scores to their errors. This is most prominently observed in Participant 3, who shows a 23.52% relative difference in confidence between correct and incorrect annotations.

5.4 Robustness against paraphrasing attacks

One simple method to bypass detectors involves using another model to paraphrase the generated lyrics. This approach changes the characteristics of the machine-generated lyrics, which can potentially deceive the detectors and reduce the number of sparse tokens.

To effectively evade these detectors, the paraphrasing must be targeted to specific sections of the lyrics to introduce the necessary changes needed to trick the detector. However, recent studies (Kumarage et al., 2023; Chakraborty et al., 2023) are using advanced paraphrasing models like Dipper (Krishna et al., 2023), which is an 11B parameters T5-based model, raising concerns on the impact of using more parameters than the original content generator and simply questioning the interest of using paraphrasing method over simply generating the lyrics with a bigger model.

To keep our experiments feasible on a consumer-grade device, we chose to use a Mistral 7B model quantized to GGUF 4 bits for paraphrasing the synthetic lyrics. We randomly selected half of the verses from each synthetic lyric and applied recursive paraphrasing (Sadasivan et al., 2024) using the specified prompt (Appendix B).

Impact on performance We observe in Table 4 that paraphrasing attacks do not degrade the performances of most detectors on machine-generated content, except for those using Shannon entropy (-11.6%). In contrast, some detectors like LLM2Vec-based ones show improved performance (+1.7%) over initial lyrics on the generated content. This improvement might be due to the paraphrasing models assigning tokens with probabilities worse than the original generation. The size of the paraphrasing models used can also be a reason for this limited impact. Using larger models from different architectures could introduce greater "uncertainty", making detection more challenging. We can also observe that embeddings-based approaches are much more impacted by this type of attack than statistical ones.

6 Conclusion

In this paper, we curated the first dataset of high-quality synthetic lyrics; we conducted a quantitative evaluation of a wide range of few-shot content detection approaches, while testing for generalization, and we complemented this with a human evaluation. Our best few-shot detector, based on LLM2Vec, outperforms previous stylistic and statistical methods in distinguishing human-written from machine-generated lyrics, generalizes well to new artists and models, and accurately detects post-generation paraphrasing. Also, these representations revealed better capabilities to capture intricate details that humans often missed. This work paves the way for future research on creative content to thoroughly investigate the generalization abilities of the detectors across a broader range of genres, artists, languages and acquisition modalities such as speech transcriptions.

The datasets, pre-processing scripts, and codes are accessible on Hugging Face[8] and GitHub[9] under Apache 2.0 license. For the real lyrics, only their titles, sources, and fingerprints will be shared to comply with the copyright policies.

[8]Hidden for double-blind.

[9]Hidden for double-blind.

7 Limitations

Our study has several limitations that should be acknowledged. Firstly, the scalability of our system to encompass a broader range of artists, genres, models, and languages may reveal limitations in the representation of the vector space. This scaling could lead to issues such as vector space collapsing, which would affect the overall performance and accuracy of the system.

Second, the rapid evolution of models poses a significant challenge. The detector's performance may become outdated quickly, complicating its use in production environments. As new models emerge, the efficacy of our current detectors may diminish, necessitating continual updates and improvements to maintain their relevance and accuracy.

Additionally, the prompting methods used to obtain lyrics from the models are frequently changing. This variability makes it difficult to consistently capture the content that a specific model can theoretically generate. The evolving nature of these methods introduces a layer of unpredictability and inconsistency, which can hinder the reproducibility and reliability of our results.

The impact of specific attributes of the lyrics, such as length, potential gender biases or verb tenses and mood, are not well understood. These attributes could influence the performance of the detectors and represent potential vectors for future types of attacks. Further research is needed to explore these aspects and understand their implications fully.

Our study's evaluation was limited to the English language. We have not assessed the detectors' effectiveness in other languages, and thus, we cannot generalize our findings beyond English. Adapting our methods to accommodate different language families would be necessary to enhance their generalizability and effectiveness in a multilingual and multi-cultural context.

Additionally, we did not examine how the genre and popularity of the lyrics might affect the detector's performance, which might significantly influence their effectiveness. Furthermore, we did not investigate deeply the paraphrasing methods and evaluate their quality. This could introduce noise and artifacts into the lyrics, potentially explaining our findings and not necessarily representing typical modifications by human creators.

Human evaluation in our study was also limited in scope. Expanding the participant pool to a more diverse socio-economic population would provide more robust and generalizable insights.

Finally, we constrained our study to models that are runnable on consumer-grade laptops under a limited number of parameters (13B parameters). While this was done to maintain feasibility, it introduces a bias, as the detectors might behave differently when scaled to larger models (70B parameters and beyond), mixtures-of-experts (Fedus et al., 2022; Jiang et al., 2024), proprietary API (OpenAI et al., 2024; Chowdhery et al., 2024), or different architectures such as Mamba (Gu and Dao, 2024).

8 Ethical Consideration

Paradoxically, revealing the inner workings of a detection system can empower malicious actors to exploit its vulnerabilities or enable them to craft content that evades detection, in order to potentially cause significant harm. This not only increases the risk of harmful content spreading, but also makes it more difficult to maintain those systems and opens the door to more sophisticated abuses.

Additionally, the risks associated with revealing the inner workings of such a detection system are fairly limited in practice due to the availability of existing literature in other domains or the rapid evolution of the approaches and continuous improvement of our systems. We also plan to enhance the robustness and adaptability of our detection algorithms to stay ahead of potential exploitation techniques. By sharing our findings, we aim to foster innovation and improvements in creative content detection systems across the NLP field.

References

Harika Abburi, Kalyani Roy, Michael Suesserman, Nirmala Pudota, Balaji Veeramani, Edward Bowen, and Sanmitra Bhattacharya. 2023. A simple yet efficient ensemble approach for AI-generated text detection. In *Proceedings of the Third Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 413–421, Singapore. Association for Computational Linguistics.

Sahar Abdelnabi and Mario Fritz. 2021. Adversarial watermarking transformer: Towards tracing text provenance with data hiding. In *42nd IEEE Symposium on Security and Privacy*.

Darius Afchar, Gabriel Meseguer-Brocal, and Romain Hennequin. 2024. Detecting music deepfakes is easy but actually hard. *Preprint*, arXiv:2405.04181.

Saadaldeen Rashid Ahmed, Emrullah Sonuç, Mohammed Rashid Ahmed, and Adil Deniz Duru. 2022. Analysis survey on deepfake detection and recognition with convolutional neural networks. In *2022*

International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, pages 1–7.

AI@Meta. 2024. Llama 3 model card.

Wissam Antoun, Benoît Sagot, and Djamé Seddah. 2024. From text to source: Results in detecting large language model-generated content. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 7531–7543, Torino, Italia. ELRA and ICCL.

Sameer Badaskar, Sachin Agarwal, and Shilpa Arora. 2008. Identifying real or fake articles: Towards better language modeling. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-II*.

Anton Bakhtin, Sam Gross, Myle Ott, Yuntian Deng, Marc'Aurelio Ranzato, and Arthur Szlam. 2019. Real or fake? learning to discriminate machine from human generated text. *Preprint*, arXiv:1906.03351.

Quentin Bammey. 2024. Synthbuster: Towards detection of diffusion model generated images. *IEEE Open Journal of Signal Processing*, 5:1–9.

Morteza Behrooz, Yuandong Tian, William Ngan, Yael Yungster, Justin Wong, and David Zax. 2024. Holding the line: A study of writers' attitudes on co-creativity with ai. *Preprint*, arXiv:2404.13165.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar Van Der Wal. 2023. Pythia: a suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Tuhin Chakrabarty, Vishakh Padmakumar, Faeze Brahman, and Smaranda Muresan. 2024. Creativity support in the age of large language models: An empirical study involving emerging writers. *Preprint*, arXiv:2309.12570.

Megha Chakraborty, S.M Towhidul Islam Tonmoy, S M Mehedi Zaman, Shreya Gautam, Tanay Kumar, Krish Sharma, Niyar Barman, Chandan Gupta, Vinija Jain, Aman Chadha, Amit Sheth, and Amitava Das. 2023. Counter Turing test (CT2): AI-generated text detection is not as easy as you may think - introducing AI detectability index (ADI). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2206–2239, Singapore. Association for Computational Linguistics.

Yutian Chen, Hao Kang, Vivian Zhai, Liangze Li, Rita Singh, and Bhiksha Raj. 2023. Token prediction as implicit classification to identify LLM-generated text. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13112–13120, Singapore. Association for Computational Linguistics.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sashank Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2024. Palm: scaling language modeling with pathways. *J. Mach. Learn. Res.*, 24(1).

Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. 2023. Simple and controllable music generation. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. 2024. Ultrafeedback: Boosting language models with high-quality feedback.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of* the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051, Singapore. Association for Computational Linguistics.

William Fedus, Jeff Dean, and Barret Zoph. 2022. A review of sparse expert models in deep learning. *Preprint*, arXiv:2209.01667.

Jack FitzGerald, Christopher Hench, Charith Peris, Scott Mackie, Kay Rottmann, Ana Sanchez, Aaron Nash, Liam Urbach, Vishesh Kakarala, Richa Singh, Swetha Ranganath, Laurie Crist, Misha Britan, Wouter Leeuwis, Gokhan Tur, and Prem Natarajan. 2023. MASSIVE: A 1M-example multilingual natural language understanding dataset with 51 typologically-diverse languages. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4277–4302, Toronto, Canada. Association for Computational Linguistics.

Albert Gu and Tri Dao. 2024. Mamba: Linear-time sequence modeling with selective state spaces.

Luca Guarnera, Oliver Giudice, and Sebastiano Battiato. 2024. Mastering deepfake detection: A cutting-edge approach to distinguish gan and diffusion-model images. *ACM Trans. Multimedia Comput. Commun. Appl.* Just Accepted.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc.

Fan Huang, Haewoon Kwak, and Jisun An. 2024. Token-ensemble text generation: On attacking the automatic ai-generated text detection. *Preprint*, arXiv:2402.11167.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts. *Preprint*, arXiv:2401.04088.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2023. A watermark for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 17061–17084. PMLR.

Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. 2020. Hifi-gan: generative adversarial networks for efficient and high fidelity speech synthesis. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Frederick Wieting, and Mohit Iyyer. 2023. Paraphrasing evades detectors of AI-generated text, but retrieval is an effective defense. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Tharindu Kumarage, Paras Sheth, Raha Moraffah, Joshua Garland, and Huan Liu. 2023. How reliable are AI-generated-text detectors? an assessment framework using evasive soft prompts. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1337–1349, Singapore. Association for Computational Linguistics.

Thomas Lavergne, Tanguy Urvoy, and François Yvon. 2008. Detecting fake content with relative entropy scoring. In *Proceedings of the 2008 International Conference on Uncovering Plagiarism, Authorship and Social Software Misuse - Volume 377*, PAN'08, page 27–31, Aachen, DEU. CEUR-WS.org.

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023. Textbooks are all you need ii: phi-1.5 technical report. *Preprint*, arXiv:2309.05463.

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*, pages 16167–16188, Singapore. Association for Computational Linguistics.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the*

60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, Dublin, Ireland. Association for Computational Linguistics.

Dominik Macko, Robert Moro, Adaku Uchendu, Jason Lucas, Michiharu Yamashita, Matúš Pikuliak, Ivan Srba, Thai Le, Dongwon Lee, Jakub Simko, and Maria Bielikova. 2023. MULTITuDE: Large-scale multilingual machine-generated text detection benchmark. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9960–9987, Singapore. Association for Computational Linguistics.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. 2023. Detectgpt: zero-shot machine-generated text detection using probability curvature. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Nikola I. Nikolov, Eric Malmi, Curtis Northcutt, and Loreto Parisi. 2020. Rapformer: Conditional rap lyrics generation with denoising autoencoders. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 360–373, Dublin, Ireland. Association for Computational Linguistics.

OpenAI. 2023. Chatgpt: Language models are few-shot learners. `https://openai.com/blog/chatgpt`. Accessed: 2024-02-10.

OpenAI et al. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Andrei Popescu-Belis, Àlex R. Atrio, Bastien Bernath, Etienne Boisson, Teo Ferrari, Xavier Theimer-Lienhard, and Giorgos Vernikos. 2023. GPoeT: a language model trained for rhyme generation on synthetic data. In *Proceedings of the 7th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 10–20, Dubrovnik, Croatia. Association for Computational Linguistics.

Xiao Pu, Jingyu Zhang, Xiaochuang Han, Yulia Tsvetkov, and Tianxing He. 2023. On the zero-shot generalization of machine-generated text detectors. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4799–4808, Singapore. Association for Computational Linguistics.

Tao Qian, Fan Lou, Jiatong Shi, Yuning Wu, Shuai Guo, Xiang Yin, and Qin Jin. 2023. UniLG: A unified structure-aware framework for lyrics generation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 983–1001, Toronto, Canada. Association for Computational Linguistics.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Md Shohel Rana, Mohammad Nur Nobi, Beddhu Murali, and Andrew H. Sung. 2022. Deepfake detection: A systematic literature review. *IEEE Access*, 10:25494–25513.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Vinu Sankar Sadasivan, Aounon Kumar, Sriram Balasubramanian, Wenxiao Wang, and Soheil Feizi. 2024. Can AI-generated text be reliably detected?

Teven Le Scao et al. 2023. Bloom: A 176b-parameter open-access multilingual language model. *Preprint*, arXiv:2211.05100.

C. E. Shannon. 1948. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423.

Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. 2024. Detecting pretraining data from large language models. In *The Twelfth International Conference on Learning Representations*.

Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, Devi Parikh, Sonal Gupta, and Yaniv Taigman. 2023. Make-a-video: Text-to-video generation without text-video data. In *The Eleventh International Conference on Learning Representations*.

Rafael A. Rivera Soto, Olivia Miano, Juanita Ordonez, Barry Chen, Aleem Khan, Marcus Bishop, and Nicholas Andrews. 2021. Learning universal authorship representations. In *EMNLP*.

Rafael Alberto Rivera Soto, Kailin Koch, Aleem Khan, Barry Y. Chen, Marcus Bishop, and Nicholas Andrews. 2024. Few-shot detection of machine-generated text using style representations. In *The Twelfth International Conference on Learning Representations*.

Michael Spanu. 2019. Toward a critical approach to the diversity of languages in popular music in the era of digital globalization. *Questions de communication*, 35(1):281–303.

Jinyan Su, Terry Zhuo, Di Wang, and Preslav Nakov. 2023. DetectLLM: Leveraging log rank information for zero-shot detection of machine-generated text. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12395–12412, Singapore. Association for Computational Linguistics.

Ben Swanson, Kory Mathewson, Ben Pietrzak, Sherol Chen, and Monica Dinalescu. 2021. Story centaur: Large language model few shot learning as a creative writing tool. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 244–256, Online. Association for Computational Linguistics.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Balamurugan Thambiraja, Ikhsanul Habibie, Sadegh Aliakbarian, Darren Cosker, Christian Theobalt, and Justus Thies. 2023. Imitator: Personalized speech-driven 3d facial animation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 20621–20631.

Yufei Tian, Anjali Narayan-Chen, Shereen Oraby, Alessandra Cervone, Gunnar Sigurdsson, Chenyang Tao, Wenbo Zhao, Yiwen Chen, Tagyoung Chung, Jing Huang, and Nanyun Peng. 2023. Unsupervised melody-to-lyrics generation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9235–9254, Toronto, Canada. Association for Computational Linguistics.

Hugo Touvron et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Andrew Trotman, Antti Puurula, and Blake Burgess. 2014. Improvements to bm25 and language models examined. In *Proceedings of the 19th Australasian Document Computing Symposium*, ADCS '14, page 58–65, New York, NY, USA. Association for Computing Machinery.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. 2023. Zephyr: Direct distillation of lm alignment. *Preprint*, arXiv:2310.16944.

Adaku Uchendu, Thai Le, Kai Shu, and Dongwon Lee. 2020. Authorship attribution for neural text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8384–8395, Online. Association for Computational Linguistics.

Pengyu Wang, Linyang Li, Ke Ren, Botian Jiang, Dong Zhang, and Xipeng Qiu. 2023. SeqXGPT: Sentence-level AI-generated text detection. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1144–1156, Singapore. Association for Computational Linguistics.

Wenhui Wang, Hangbo Bao, Shaohan Huang, Li Dong, and Furu Wei. 2021. MiniLMv2: Multi-head self-attention relation distillation for compressing pre-trained transformers. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2140–2151, Online. Association for Computational Linguistics.

Kangxi Wu, Liang Pang, Huawei Shen, Xueqi Cheng, and Tat-Seng Chua. 2023. LLMDet: A third party large language models generated text detection tool. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2113–2133, Singapore. Association for Computational Linguistics.

Zhizheng Wu, Junichi Yamagishi, Tomi Kinnunen, Cemal Hanilçi, Mohammed Sahidullah, Aleksandr Sizov, Nicholas Evans, Massimiliano Todisco, and Héctor Delgado. 2017. Asvspoof: The automatic speaker verification spoofing and countermeasures challenge. *IEEE Journal of Selected Topics in Signal Processing*, 11(4):588–604.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. 2024. WizardLM: Empowering large pre-trained language models to follow complex instructions. In *The Twelfth International Conference on Learning Representations*.

Yongyi Zang, You Zhang, Mojtaba Heydari, and Zhiyao Duan. 2024. Singfake: Singing voice deepfake detection. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. 2024. Tinyllama: An open-source small language model. *Preprint*, arXiv:2401.02385.

Yipin Zhou and Ser-Nam Lim. 2021. Joint audio-visual deepfake detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14800–14809.

Biru Zhu, Lifan Yuan, Ganqu Cui, Yangyi Chen, Chong Fu, Bingxiang He, Yangdong Deng, Zhiyuan Liu, Maosong Sun, and Ming Gu. 2023a. Beat LLMs at their own game: Zero-shot LLM-generated text detection via querying ChatGPT. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7470–7483, Singapore. Association for Computational Linguistics.

Junchen Zhu, Huan Yang, Huiguo He, Wenjing Wang, Zixi Tuo, Wen-Huang Cheng, Lianli Gao, Jingkuan Song, and Jianlong Fu. 2023b. Moviefactory: Automatic movie creation from text using large generative models for language and images. In *Proceedings of the 31st ACM International Conference on Multimedia*, MM '23, page 9313–9319, New York, NY, USA. Association for Computing Machinery.

Sijin Zhu, Zheng Wang, Yuan Zhuang, Yuyang Jiang, Mengyao Guo, Xiaolin Zhang, and Ze Gao. 2024. Exploring the impact of chatgpt on art creation and collaboration: Benefits, challenges and ethical implications. *Telematics and Informatics Reports*, 14:100138.

A Prompt template

Figure 1 displays the prompt template used to generate lyrics with 3-shot in-context learning based on human-written lyrics:

3-shot Lyrics Generation Template

Example 1:

`{{lyrics 1}}`

Example 2:

`{{lyrics 2}}`

Example 3:

`{{lyrics 3}}`

Lyrics rules:

- The lyrics should be structure in optional stanzas like “Verse”, “Chorus” and “Bridge”

- The beginning of each line should start with a capital letter.

- Do not use repeat tags to signify if a line or stanza is repeated. Instead, write each line or stanza however many times it is said.

- Do not write out any sounds that are heard in the song, like “gunshot”, “clap”, “horn”, etc.

- Remove all labels such as [Talking], Speaking, or (Whispering).

- Any word cut short should have one apostrophe in place of the missing letters. For example: givin’, livin’.

- Slang is acceptable but the artist must pronounce it that way. Slang should only be used if the word sounds differently than the grammatically correct word. For example, “for shizzle” can be used but “becuz” should be spelled “because”.

- Exaggerations should be cut down to the original word or punctuation. For example, “ohhhh” should be “oh” and “bang!!!!!” should be “bang!”

- Background vocals should be placed on the same line they’re said but in parentheses. For example, “I’m a survivor (What, what)”

- Prevent using too much background vocals

Generate a new lyrics based on the style of what "`{{artist name}}`" is doing and don’t mention me the fact that the lyrics is offensive:

Figure 1: 3-shot lyrics generation template.

B Paraphrasing Prompt Template

Figure 2 display the paraphrasing prompt template used to perform the paraphrasing attack of the detectors:

Paraphrasing Template

According to the full lyrics:

`{{full lyrics}}`

Paraphrase and modify the following verse to make it sound like human (only one generated verse is allowed and no explanation):

`{{paragraph}}`

Output:

Figure 2: This template is used for the paraphrasing of the generated lyrics.

C Participants confidence score

The Figure 3 list of confidence score options and their descriptions provided to the participants:

Confidence scores options

1 = Willing to defend my annotation, but it is fairly likely that I missed some details.

2 = Pretty sure, but there’s a chance I missed something. Although I have a good feel for this area in general, I did not carefully check the lyrics details.

3 = Quite sure. I tried to check the important points carefully. It’s unlikely, though conceivable, that I missed something that should affect my annotation.

4 = Positive that my annotation is correct. I read the lyrics very carefully.

Figure 3: List of confidence scores options and their descriptions.

D Inter-participants agreement

Participant ID	κ	$\mathcal{G}$	Agreement
Participant 1 & 2	3.53	15.47	54.29
Participant 1 & 3	29.81	43.75	68.57
Participant 1 & 4	35.46	41.04	68.57
Participant 2 & 3	17.85	22.28	60.00
Participant 2 & 4	-9.29	-7.78	45.71
Participant 3 & 4	30.52	32.80	65.71
Average	17.98	24.59	41.42

Table 5: Inter-participants agreement statistics. κ is referring to Kappa Cohen and $\mathcal{G}$ to Gwet’s AC1.

E Hyperparameters

Table 6 lists all the hyperparameters used during the lyrics generation process to ensure reproducibility:

Parameter	Value
temperature	0.8
top_k	40
top_p	0.9
num_predict	2048
quantization	Q4_0
seed	42

Table 6: Hyperparameters for the lyrics generators LLMs.

We used 3 NVIDIA RTX A5000 24GB gpus for our experiments during approximately 30 hours of computation.

F Min-K % Prob impact of K

The K value is impacting a lot the performance as seen in the Table 7. In the case of our specific data, we observe an optimal K value at 10.

	Recall	
Min-K% (%)	Normal	Paraphrased
5	77.00	81.31
10	79.23	82.74
20	73.48	77.32
30	64.31	69.08
40	59.00	65.05
50	57.01	63.07
60	53.38	61.00
70	52.69	58.59
80	52.86	58.96

Table 7: Overall recall on the test set (with and without paraphrasing attack) for the Min-K% Prob detector according to the selected K value.

G Lyrics spit out during generation

To ensure our generative models for creating machine-generated lyrics don't merely reproduce the provided few-shot examples, we conduct an analysis of the generated lyrics. We index all the human-written lyrics used to condition the model's generation using BM25 (Trotman et al., 2014) representation and then compute the similarity with the generated lyrics to determine the ranking of the few-shot examples and see if the outputs are resembling very much to the references.

	% Hits	
Rank	% Hits	Cumulated % Hits
1	2.28	2.28
2	1.05	3.34
3	0.83	4.17
3 to 5	1.37	5.55
5 to 10	2.57	8.12
10 to 20	3.94	12.06
20 to 50	7.79	19.86

Table 8: Proportion of hits by range of ranks between the generated lyrics and the given 3-shot examples.

As we can see in the Table 8, the 3-shot lyrics used for conditioning the generation aren't well ranked in overall. This significantly reduces the doubt that the generated lyrics are heavily based on the set of lyrics provided as input for conditioning their generation. However, it does not guarantee that certain parts, such as verses, are not entirely copied from the few-shot examples.

H Human's annotation feedback

We request the participants to answer three questions after completing the annotation of the 70 lyrics to gather their feedback on the task they have performed. All the participant's feedback are listed in the following Figure 4:

Participant's Feedback

Q1: Can you write me a short explanation of what do you refer to when you were labeling the lyrics ? Which characteristics have motivated your choices ?
Answer P1: I was looking to multiple characteristics, such as if the refrain is every time the same or not, the rhythms at the end of the sentences, the sparsity of the words used at the beginning of the sentences or the overall structure of the lyrics.

Answer P2: I expected lyrics to be generated if there was too much repetition, excessive punctuation (particularly too many commas within the verses), very few rhymes, or if the length of the lyrics was excessively long.

Answer P3: Generally, I started by looking at the structure of the lyrics. Which paragraph corresponds to the choruses, whether the verses are of similar length or not, and whether there is a visible structure that stands out. If no particular structure stood out, I focused on the coherence of the lyrics. If there was a noticeable structure, I also looked at the rhymes and the progression of the story verse by verse. If the rhymes were poorly done/strange or of uneven quality, if the verses were too unbalanced, if lyrics from the verses were repeated in the choruses, or if there was not much difference between a verse and a chorus, I tended to consider it as machine-generated.

Answer P4: The main point for me is the song's structure. Machine-generated lyrics often have a more poetic than lyrical structure. The variations of the chorus were another key indicator, in particular, machine-generated lyrics tend to create many different versions. Another hint for me was the use of counterpoints (usually in parentheses), which machine-generated lyrics tend to overuse. Finally, whenever the topic of the lyrics was explicit, it was definitely a human-written lyric, since machine are not conditioned to generate such content.

Q2: Have you been able to recognize one or more songs during the annotation ?
Answer P1: Yes, one song "Red" by Taylor Swift.

Answer P2: 1 song from Taylor Swift

Answer P3: I had the feeling that I recognized two other songs. In those cases, I gave a rating of maximum confidence.

Answer P4: Yes, two.

Q3: Do you consider it as difficult task and why ? (short answer only)
Answer P1: Yes, it is difficult to get confident on some lyrics since I am not used to focusing on the lyrics when listening to a song.

Answer P2: Yes, especially the rap and hip hop songs. The lyrics were very convincing and often I felt like guessing the answer with no real idea of what to choose.

Answer P3: I found this task relatively difficult (as shown by my confidence score), so yes.

Answer P4: Yes. Most of the topics are coherent and follow a natural story telling. Rhymes are also nice. So I needed to focus on other aspects.

Figure 4: The participant's feedback on the human evaluation process.

I Participants confidence scores

Participant ID	Error	Correct	Relative Δ
Participant 1	3.35	3.43	2.35 %
Participant 2	2.07	2.18	5.17 %
Participant 3	1.95	2.47	23.52 %
Participant 4	2.37	2.42	2.08 %

Table 9: Confidence scores averaged when errors are committed.