Evaluation of Artificial Intelligence Generated Content Detection Strategies

Luke M. Unterman, B.S. in Computer Science[1]
[1]**Virginia Commonwealth University, Richmond, Virginia, United States of America**

Abstract

Artificial Intelligence Generated Content (AIGC) has exploded in popularity within recent years, drawing significant attention from both academia and industry. This has largely coincided with the advent of modern large language models based on the Transformer architecture, some of which use billions of different parameters. ChatGPT, released by OpenAI, in particular has dramatically influenced the rate at which regular people interact with AIGC. ChatGPT excels at translating natural language to code, completing extremely masked texts, and generating stories given user-defined elements and styles. Because ChatGPT is openly accessible to the public, there are concerns especially regarding plagiarism and the occasional inaccuracy of its responses within professional and educational contexts. This paper attempts to alleviate that concern associated with AIGC by evaluating AIGC detection techniques such as statistical and linguistic analysis to aid researchers and developers in creating effective AIGC (primarily ChatGPT) detection tools.

1 Introduction

Artificial Intelligence Generated Content (AIGC) has exploded in popularity within recent years, drawing significant attention from both academia and industry[1]. This has largely coincided with the advent of modern large language models based on the Transformer[2] architecture, some of which use billions of different parameters. This Transformer model, which is based solely on attention mechanisms, can be trained significantly faster than architectures based on recurrent or convolutional layers, as it achieves "better BLEU [BiLingual Evaluation Understudy] scores than previous state-of-the-art models" such as ByteNet and ConvS2S on different translation tasks at a "fraction of the training cost"[2]. The Transformer works by eschewing the dominant sequence transduction models based on complex recurrent or convolutional neural networks that include an encoder and a decoder and instead relies entirely on an attention mechanism to draw global dependencies between input and output[2]. These improvements in the efficiency with which Transformer models can be trained have allowed for generative pre-trained Transformer models like ChatGPT-3 to effectively solve tasks such as question answering, semantic similarity assessment, entailment determination, and text classification.

ChatGPT, released by OpenAI,[3] in particular has dramatically influenced the rate at which regular people interact with AIGC. ChatGPT is a large language model based on the Transformer and is fine tuned from the GPT-3.5 series with Reinforcement Learning from Human Feedback. ChatGPT in particular excels at translating natural language to code, completing extremely masked texts, and generating stories given user-defined elements and styles. It also completely excels at basic NLP tasks like classification, entity extraction, and translation.[4] Prior iterations of GPT models like GPT-3 typically failed at properly responding to human queries, although ChatGPT seems to have improved greatly in human interactions.

However, because ChatGPT is so effective at generating high-quality AIGC and is completely open to the public, there has been growing concern within the educational domain regarding the potential negative implications of high-quality AIGC being accessible to students. One such concern is plagiarism, as ChatGPT is capable of generating graduate-level responses tailored to any given question. Wharton professor Christian Terwiesch determined that the ChatGPT generated responses to the final exam of a typical MBA core course (operations management) would have received a grade of B/B- when following the exam's typical rubric[5]. Another concern associated with the accessibility of this AIGC is that ChatGPT's responses are not totally infallible. While ChatGPT can be incredibly effective at generating highly specific text-based responses, it can occasionally make surprising mistakes in relatively simple calculations at the level of the 6th grade[5]. StackOverflow has banned ChatGPT-generated content completely, as the average rate of getting correct answers from ChatGPT[4] is "too low."

Naturally, as a result of these concerns, there has been a considerable effort to create programs that can accurately detect content generated by ChatGPT or AIGC in general. Wang et al. wrote a paper entitled 'Evaluating AIGC De-

tectors on Code Content' which presented the first empirical study evaluating the performance of AIGC detectors such as GPT-2 Detector[6], RoBERTa-QA[7], DetectGPT[6], and GPTZero[8] on code-related content generated by ChatGPT. This paper diverges from Wang et al.'s paper, as its purpose is to compile information regarding AIGC detection techniques such as linguistic and statistical analysis using a variety of features and Machine Learning (ML) classification models. The goal of this paper is evaluate the effectiveness of these techniques in an attempt to aid other researchers and developers in determining how they should approach creating AI content detection tools. This paper also includes a briefly recounted history of the evolution of different ML models such as convolutional and recurrent neural networks and transformers in order to clarify their differences.

2 History

Prior to 2016, many core NLP techniques have been dominated by machine learning approaches that used linear models such as support vector machines (SVMs) or logistic regression which have been trained over high dimensional yet sparse feature vectors. However, as of 2016, the NLP field started to switch from linear models to non-linear neural network models over dense inputs[9]. Recursive and recurrent architectures in particular were capable of producing state-of-the-art results for both sentiment classification and question answering, as they can work with sequences and trees while preserving structural information[9]. One particular problem with recursive neural networks (RNNs) is that during training, components of the gradient vector can grow or decay exponentially over long sequences[9]. Long short-term memory networks (LSTMs) were designed to address this issue surrounding the learning of long-term dependencies, as they maintain a separate memory cell inside it that updates and exposes its contents only when deemed necessary[10].

The Transformer architecture was then invented in 2017 by Vaswani et al. and his team within Google Brain, which eschews the dominant sequence transduction models based on complex recurrent or convolutional neural networks that include an encoder and a decoder and instead relies entirely on an attention mechanism to draw global dependencies between input and output[2]. The transformer model in particular allows for significantly more parallelization and can reach a new state of the art in translation quality in as little as twelve hours on eight P100 GPUs. Devlin et al. then expanded upon the Transformer by introducing BERT[11] which stands for Bidirectional Encoder Representations from Transformers. This version of the Transformer is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditional on both left and right context in all layers, and it improved the General Language Understanding Evaluation (GLUE) score[12] of the original Transformer from 72.8% to 80.5% .

Then, in 2018, Radford et al. and his team at OpenAI created a generative pre-trained transformer model (GPT) using the Transformer architecture, which is the original prototype of the ChatGPT-3.5 model which is the primary focus of this paper[3]. This GPT model utilized a semi-supervised approach for language understanding tasks they developed using a combination of unsupervised pre-training and supervised fine-tuning. Furthermore, the GPT model uses a multi-layer Transformer decoder for the language model, which applies a multi-headed self-attention operation over 'the input context tokens followed by position-wise feedforward layers to produce an output distribution over target tokens.'[3] OpenAI ultimately developed a model which steadily increases over training, giving credence to the idea that 'generative pre-training supports the learning of a wide variety of task relevant functionality' and demonstrating how a pretrained model could outperform the Transformer model without pre-training via an increase in performance across all evaluation measures of 14.8%. GPT-2, RoBERTa[7], and GPT-3.5 were subsequently released in 2019, 2019, and 2020 respectively which further optimized the GPT and BERT models. GPT-2 and GPT-3 were optimized and pre-trained on WebText and Common Crawl respectively, and RoBERTa used dynamic Masked Language Modeling, removed the Next Sentence Prediction objective, employed a BPE tokenizer, and used better hyperparameters.

3 Detection Techniques

In this section, I explore the different approaches of detecting AIGC, such as linguistic and statistical analysis and divide each of the approaches into different subsections. The datasets used, pre/post-processing, features used, evaluation methodologies, and results will all be described.

3.1 Linguistic Analysis

Within the field of NLP, linguistic analysis is an essential component required to gain understanding of the corpus with which you are analyzing. Linguistic analysis is a method used to analyze text based on the principles of linguistics, involving the examination of both grammatical and syntactical features of text. This can include sentence structure, word choice, punctuation, parts of speech distribution, word density, uniqueness, dictionary length, and more. The most commonly used large language models today can output text that is convincing, but may produce results that are extremely syntactically similar or may use certain types of words like NNPs more frequently than humans typically do. AIGC may also be limited in its ability to produce overly biased text or text that exhibits variation in either sentence structure or style.

3.1.1 Bhatt & Rios

In their paper entitled 'Detecting Bot-Generated Text by Characterizing Linguistic Accommodation in Human-Bot Interactions', Bhatt & Rios use linguistic analysis in an attempt to alleviate potential concerns regarding the abuse of bots to spread misinformation through AIGC detection[13]. The detection of bots online is well studied, and relies on two forms of information: behavior and content. Behavior relates to the frequency with which bots post, and content involves using the bot's text directly. In their study, Bhatt & Rios focus on Communication Accommodation Theory, which is used to study language use in various domains to understand human behavior. Within the scope of their study, they use linguistic accommodation to better understand human-bot interactions. They also attempt to explore two types of datasets, including one where researchers instruct participants to interact with bots and humans in the same way, and one where participants are instructed to converse with a bot.

In order to distinguish between human-human and human-bot interactions, Bhatt & Rios attempted to develop a classifier $f(D)$ which maps to a class in the set $T = \{\text{human-human, human-bot}\}$[13]. Obviously, the human-human class refers to an interaction between two humans, and the human-bot class refers to an interaction between a human and the text generated by a bot. They formulated two bot detection datasets for this task using three well-known bot datasets, including ConvAI2, WOCHAT, and Daily-Dialog. They used four datasets from ConvAI2 for their experience, containing both human-human interactions and human-bot interactions. WOCHAT and Daily-Dialog were used to form a Control Dataset, which contained interactions between humans and bots such that humans were not instructed to converse with the bots as if they were human.

They processed the data into three groups: Unpaid (U), Paid (P), and Control (C). The U dataset consists of PERSONA-CHAT, INTERMEDIATE, and VOLUNTER ConvAI2 datasets. The P dataset consists of PERSONA-CHAT and TOLOKERS. The C dataset consis of IRIS, TICK-TOCK, and DailyDialog [citation] Each dataset was divided into 70%, 10%, and 20% training validation and test splits, respectively[13].

The different features they used for the linguistic analysis of the two datasets they created were split into groups, as they either were content-based (aka features describing "what" humans and bots say in their interactions") and style-based (aka features encoding "how" humans and bots speak in their interactions). The content features they used were bag-of-words (TF-IDF weighted unigrams from a dialog) and embedding features (dialogue encoded using BERT). The stylistic features they used were Linguistic Inquiry and Word Count (LIWC) and linguistic accommodation. The LIWC feature describes positive emotion, cognitive, and social processes. Linguistic alignment measures the proportionate increase in the likelihood of a word being used when it has already been used previously in conversation. They trained their model using a Logistic Regression from the Scikit-Learn package for the content feature sets. For the stylistic features, they used the Random Forest classifier from the Scikit-Learn package[13].

They ultimately found that BERT-based models were able to outperform the Bag-of-Words models on average for the content features. Furthermore, as seen above in Table 3, content-based features were determined to be more predictive/discriminative for bot detection when training and testing on the same train-test splits from the same dataset. For the stylistic features, they determined that combining both Accommodation and LIWC features from both the Human and Bot is better than using either feature set individually. The three major findings from their research are that it is 1) more informative to analyze human language in each conversation than the bot's content, 2) the generalization performance of BERT drops compared to using bag-of-words, and 3) accommodation features outperform all other

individual feature sets with an AVG bot-detection Macro F1 of 0.584[13].

3.1.2 Guo et al.

Guo. et al also devised a classification model that used linguistic analysis features to differentiate between human responses and ChatGPT-based responses, as their goal for the study was to be able to determine the capabilities of ChatGPT in comparison to human experts. They collected nearly 40 thousand questions and their corresponding answers from both human experts and ChatGPT covering a wide range of domains and named it the Human ChatGPT Comparison Corpus (HC3) dataset. In terms of the different responses collected from the HC3-English dataset, 58,546 answers were collected from humans from datasets such as the $reddit_eli5$ dataset and 26903 answers were collected from ChatGPT[4].

Using the HC3 dataset, Guo et al. conducted comprehensive human evaluations as well as linguistic analysis on both human and ChatGPT generated answers. They divided said human evaluations into the Turing test and the Helpfulness test. The Turing test is a test of a machine's ability to exhibit intelligent behavior that is indistinguishable from two groups. They invited 17 volunteers (8 experts on ChatGPT output and 9 amateurs) to evaluate the Turing test. Four types of evaluations were used: Expert Turing Test/Paired Text, Expert Turing Test/Single Text, Amateur Turing Test/Single Text, and a Helpfulness test which attempted to provide insight into the helpfulness of the different responses[4].

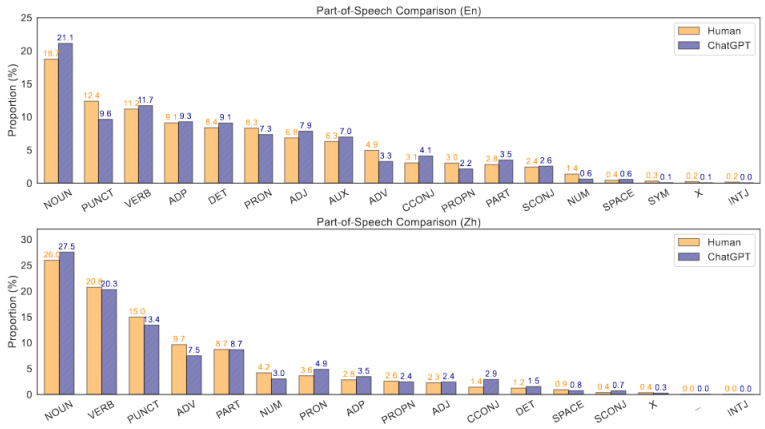

Figure 1: Part-of-Speech distribution comparison between ChatGPT and human answers. Results are sorted by POS proportion of human answers. The upper figure is for the HC3-English dataset and the lower is for the HC3-Chinese dataset.

As seen in Figure 1. above, Guo et al. devised relatively simple metrics to evaluate the human-evaluated performance of ChatGPT and human generated content. They found the average length (num words in question), vocab size (unique words), and word density (number of different words used in same length of text) of the human and AI-generated responses. Human answers typically provide shorter, yet denser responses than ChatGPT does. Other linguistic analysis they conducted measures the POS composition of the different texts. Through their testing, they determined that ChatGPT uses more NOUN, VERB, DET, ADJ, AUX, CCONJ, AND PART words, while using less ADV and PUNCT words[4].

They also used dependency parsing and sentiment analysis to measure the dependencies between the words used in the different corpora and the positive, neutral, or negative sentiments associated with different words, respectively. Their findings regarding the dependency parsing were that ChatGPT has much longer distances between punct and dep relations, which they concluded was a result of ChatGPT's comparatively longer sentences. ChatGPT often also uses more conjunctions than humans to make the content more logical. For their sentiment analysis findings, they concluded that ChatGPT generally expresses more neutral sentiments than humans. Humans also express significantly more negative emotions than ChatGPT does[4].

Guo et al. determined that the pair-expert evaluation type was more discriminative at detecting ChatGPT-generated content than by using the single-expert evaluation type. Interestingly, ChatGPT's answers were considered to be more helpful than human responses in more than half of the questions[4].

Through inviting volunteer testers of different professional backgrounds (professional vs. amateur) to evaluate the Turing test of the question/answers pairs. Guo et al. found that although ChatGPT answers could be easily distinguished from human answers when testers were given a comparison pair, ChatGPT's answers were generally considered to be more helpful to humans. Guo et al. implemented three different methods from classic machine learning and deep machine learning to determine the most effective AIGC content classification model[4].

For the classic machine learning method, they used a logistic regression model trained on the GLTR Test-2 features. For the deep-learning method, they used a deep classifier for single-text detection and a deep classifier for QA detection.[citation]. These deep classifiers were both RoBERT-based. Of the different classification algorithms that Guo et al. used when finding a model to best detect ChatGPT answers, they determined that their strong pre-trained Transformer model RoBERTa was most effective at detecting ChatGPT answers when given the full text. Their conclusions were that the RoBERTa-based detectors were ultimately most effective as they performed better than GLTR in terms of F1 scores, were not affected by indicating words, and were more effective in handling Out-Of-Distribution scenarios[4].

3.2 Statistical Analysis

Unlike the previous artificial intelligence detection strategies, Nguyen-Son et al.[14] detail how one can use statistical analysis to differentiate computer-generated text from human-generated text. Statistical analysis refers to the examination of patterns regarding certain statistical properties in AIGC. This can involve the analysis of the Zipfian distribution of words, sentence lengths, word frequencies, n-gram distributions, and other statistical measures. The term Zipfian distribution refers to Zipf's law, which states that the most frequent word in human-written text has approximately twice the frequency of the second most frequent word, nearly three times that of the third most frequent word, and so on[14].

In Nguyen-Son et al.'s paper entitled 'Identifying computer-generated text using statistical analysis', Nguyen-Son et al. evaluate the differences in statistical features such as the presence of complex phrases and the consistency of documents at the sentence level using phrasal verbs to differentiate human and computer-generated text in documents. To be specific, their method for detecting computer-generated text using statistical features involves the evaluation of the word frequency distribution in original and computer-generated documents to see if the texts follow a Zipfian distribution, the extracting of complex phrases from the text including idiomatic phrases, cliched phrases, and ancient phrases, the measuring of the consistency of documents at the sentence and paragraph level, and the combining of these statistical features to create a classification model. In order to extract these features, they translated 100 English books into Finnish using Google Translate and used these documents as their datasets[14] .

In order to extract the frequency features, Nguyen-Son et al. extracted a linear regression line feature such that a lemma distribution is calculated and used to estimate linear extraction function $f = ax + b$, and they extracted information loss including square root feature R^2 and cost value feature. In order to extract complex phrase features, Nguyen-Son et al. extracted idiomatic phrases, cliched phrases, ancient phrases, and dialect phrases. Finally, Nguyen-Son et al. extracted text consistency features by quantifying both phrasal verb features and coreference resolution features. More specifically, they used the Stanford NLP library to identify PRT tags to phrasal verbs and they calculated the coreference features by dividing the number of coreference resolution relationships (found also using the Stanford NLP library) by the number of words in the document[14] .

Nguyen-Son et al. evaluated these features using 10-fold cross validation to create classifiers, opting to use logistic regression and Support Vector Machines (SVMs). From their results, they determined that the most important feature was the frequency of ancient-phrases. This essentially demonstrates how machine translation tends to use simple words and phrases. They SVM algorithm they used had the highest performance with an accuracy of 89% and an EER score of 10.2%. However, they noticed that using a combination of all of the aforementioned features performed the best with an accuracy of 98%[14] .

Other statistical detection methods involve distinguishing between human-generated text and computer-text based on

the "salad phenomenon"[15]. The salad phenomenon is when each phrase of computer-generated text is grammatically correct, but when put together, they are incorrect in terms of collocation. This phenomenon is estimated in their model using an N-gram language model for continuous word sequences cases and sequential pattern mining for isolated word cases. They determined that this method works well not only for documents but for sub-document levels such as sentences and phrases. Other detection methods use text-similarity based approaches. Labbe et al.[16] measured the inter-textual similarity of academic papers using word distributions. It works by looking at technical terms and phrases in corresponding fields, although this method is unsuitable for detecting computer-generated text in the general domain.

4 Compare and Contrast

In this section, the features, analysis types, classification models, and model performance will be compared. Upon looking at Table 1 below, one can immediately spot a pattern differentiating the classification models used for statistical analysis and used for linguistic analysis. However, I suspect that this is largely the result of the difference in publishing dates between the articles (the linguistic analysis models are from 2021 and 2023 respectively, while the statistical analysis models are from 2017 and 2013 respectively). As explained in the history section of this literature review, the incredibly influential Transformer-based models were not invented until 2017, so it would make sense for classical classification models like Support Vector Machines to be most effective in the models which used statistical analysis. This is somewhat redundant, but it is also clear that the statistical models used features which analyzed the general structure of their datasets (like word frequency distribution and document density) while linguistic models analyzed features which analyzed the individual qualities of the words used in their datasets (POS tags, average length, sentiment analysis).

Author	Analysis Type	Feature Used	Classification Model
Bhatt & Rhios	Linguistic	Bag-of-words, LIWC	BERT-based
Guo et al.	Linguistic	POSTags, sentiment analysis	RoBERTa
Nguyen-Son et al.	Statistical	word frequency distribution, complex phrases	SVMs
Arase & Zhou Abstract	Statistical	completeness of non-contiguous phrases	SVMs

Table 1: Author, analysis type, features used, and best performing classification model

Although perfect comparisons between the two linguistic analysis models cannot be drawn as a result of different AIGC datasets being used (ConvAI2 vs ChatGPT), both Bhatt & Rios and Guo et al. demonstrate how a combination of linguistic features such as bag-of-words and POS-tags can be used to effectively distinguish between AIGC. The main difference between the results of these two models which use linguistic features are that the most discriminative features were human LIWC and accommodation features for the Bhatt & Rios model achieving a Macro F1 of 0.660, while the dataset of raw-sentences (raw-sent) using the combination of features in Table 1 proved to be most accurate at detecting ChatGPT-generated text when compared to Human Expert-generated texts with an average F1 score of 98.78%.

The two statistical models made by Nguyen-Son et al. and Arase & Zhou are actually fairly similar in their approach. Nguyen-Son et al. use a combination of word frequency distribution, complex phrases, and document consistency features resulting in an accuracy of 98.0% and an equal error rate of 2.9%.[14] However, Arase & Zhou[15] use statistical machine translation systems and focus on the phrase salad phenomenon to detect machine-translated texts from a large-scale Web-mined text. The models produced in both articles use Support Vector Machines to determine the likelihoods of machine-translated and human-generated texts and include Google Translate translations in their datasets. Therefore, I feel comfortable in stating that Nguyen-Son et al.'s approach improved on Arase & Zhou's, as its accuracy in detecting machine-translated texts surpassed Arase & and Zhou's by 2.2% (98.0% vs 95.8%).[14,15]

Although I cannot compare the performances of the linguistic models vs the statistical models, it is clear that the linguistic models are more effective. Said linguistic analysis models both are BERT-based classification models and share characteristics with the most common GPT-detection models currently available to the public. For example, GPT2-Detector is a commonly used AIGC detection tool and utilizes linguistic analysis as well as RoBERTa to detect GPT-2 generated content at a rate of 95%.[1]. From what I can tell, most of the AIGC-detectors used in Wang et al.'s

study (which empirically evaluated different GPT-content detectors) utilized linguistic analysis.

5 Conclusion

In this literature review, I briefly recount the evolution of Large Language Models starting with convolutional neural networks, describe the main differences and details associated with implementing AIGC-detectors which use either statistical or linguistic analysis, and perform a brief evaluation of the performances of different models which employ statistical and linguistic analysis. I determined that AIGC-models which use linguistic analysis are most likely to be more effective given the current state of publicly available ChatGPT/AIGC- detectors. The field of AIGC-detection is still burgeoning, as the first ever empirical evaluation of said AIGC-detectors was published by Wang et al. in 2023. Furthermore, all of the detectors featured within this literature review attempted to classify bodies of text; the lack of AIGC-detectors which specifically attempt to classify machine-generated code is one of the significant gaps within the NLP literature. Developing a model to detect code generated by Artificial Intelligence could be a potential area of future study. I would also recommend developing a model which compared the effectiveness of features used for the linguistic analysis of text as well as the statistical analysis of test to further the differences in effectiveness between the two methods.

References

1. Jian Wang, Shangqing Liu, Xiaofei Xie, and Yi Li. Evaluating aigc detectors on code content, 2023.
2. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.
3. Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.
4. Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. How close is chatgpt to human experts? comparison corpus, evaluation, and detection, 2023.
5. Christian Terwiesch. Would chat gpt get a wharton mba? a prediction based on its performance in the operations management course, 2023.
6. Openai: Gpt-2 detector, 2019.
7. Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019.
8. Gptzero, 2022.
9. Yoav Goldberg. A primer on neural network models for natural language processing, 2015.
10. Ralf C. Staudemeyer and Eric Rothstein Morris. Understanding lstm – a tutorial into long short-term memory recurrent neural networks, 2019.
11. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.
12. Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding, 2019.
13. Paras Bhatt and Anthony Rios. Detecting bot-generated text by characterizing linguistic accommodation in human-bot interactions. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3235–3247, Online, August 2021. Association for Computational Linguistics.
14. Hoang-Quoc Nguyen-Son, Ngoc-Dung T. Tieu, Huy H. Nguyen, Junichi Yamagishi, and Isao Echizen. Identifying computer-generated text using statistical analysis. In *2017 Asia-Pacific Signal and Information Processing*

Association Annual Summit and Conference, pages 1504–1511, United States, February 2018. Institute of Electrical and Electronics Engineers (IEEE). 2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA ASC 2017 ; Conference date: 12-12-2017 Through 15-12-2017.

15. Y. Arase and M. Zhou. Machine translation detection from monolingual web-text. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics,*, pages 1597–1607, 2013.

16. Cyril Labbé and Dominique Labbé. Duplicate and fake publications in the scientific literature: how many SCIgen papers in computer science? *Scientometrics*, pages 10.1007/s11192–012–0781–y, June 2012.