Detecting ChatGPT-Generated Text with GZIP-KNN: A No-Training, Low-Resource Approach

Matthias Berchtold[*] and **Sandra Mitrović**[†] and **Davide Andreoletti**[‡] and **Daniele Puccinelli**[‡] and **Omran Ayoub**[‡]

[*] University of Applied Sciences and Arts of Southern Switzerland (SUPSI), Switzerland

[†] Dalle Molle Institute for Artificial Intelligence (IDSIA), USI/SUPSI, Switzerland

[‡] Institute of Information Systems and Networking (ISIN), SUPSI, Switzerland

Abstract

Text classification is a fundamental Natural Language Processing task that is mostly addressed with resource-intensive transformer architectures. Researchers are continuously investigating lightweight alternatives without compromising predictive efficacy. A lightweight alternative called *Gzip-KNN* that combines the compression capability of Gzip with the K-nearest neighbors (KNN) classifier has been recently proposed. In this paper, we investigate the potential of Gzip-KNN for the detection of AI-generated text, notably ChatGPT-generated content. We compare its performance to several streamlined machine learning models such as Logistic Regression, eXtreme Gradient Boosting, and Gated Recurrent Unit. Our evaluation considers predictive accuracy, training duration, and inference speed, all while adjusting the available data in in- and out-of-domain contexts. Our experimental results highlight that *Gzip-KNN* achieves high predictive performance, often surpassing other models, especially when operating on a limited dataset for inference. Nonetheless, its extended inference time restricts its utility in time-sensitive scenarios. Intriguingly, *Gzip-KNN* manages to match the performance of other tested approaches even when utilizing a very limited percentage of the available data.

Keywords ChatGPT, Generative Language Models, Bots, GZIP

1 Introduction

The task of text classification, i.e., the categorization of a text into predefined classes, is fundamental in the domain of Natural Language Processing (NLP). The general approach involves designing a function that maps texts to their corresponding classes. Such function is generally obtained with supervised machine learning. Specifically, supervised training is performed by tuning the parameters of the models to minimize the error between estimated and ground truth classes. The complexity of the training process highly depends on the number of model parameters, which typically ranges from a few to millions.

Conventional approaches for text classification rely on complex models such as neural networks and in particular, transformers-based architectures, which are characterized by millions of parameters. These models yield remarkable predictive performance at the cost of a high training complexity. Indeed, training these models is expensive in terms of the amounts of data required, computational power, and training time (Chollet, 2017; Thompson et al., 2020). Therefore, employing such models for text classification may be an overkill. Rather than relying solely on large models, there is a growing interest in rediscovering lightweight approaches that can match the predictive accuracy of more complex models while requiring less computational power and training data (Fournier et al., 2023; Gururangan et al., 2019; Pan et al., 2019).

Recently, Jiang et al. (Jiang et al., 2023) proposed a lightweight methodology for text classification based on the combination of data compressing techniques (the *Gzip* compressor) and a low-complexity classifier (KNN, i.e., the K-nearest neighbors algorithm).

The proposed approach is referred to as *Gzip-KNN*, and is discussed in detail in Section 3. One of the distinctive traits of *Gzip-KNN* is the high computational efficiency due to its simple underlying components and non-parametric nature. In fact, the absence of tunable parameters drastically simplifies the training process. The intuition behind *Gzip-KNN* is that samples belonging to the same class are inherently more regular compared to samples from different classes. Hence, a lossless compression technique, such as the well-known *Gzip* algorithm, can be used to obtain representations that capture this intrinsic regularity. Subsequently,

the representation of the sample undergoing classification is compared with the representations of the training samples using a novel distance metric. This process yields a distance matrix, which serves as the input for a k-nearest-neighbor classifier. In their study, the authors compare the predictive performance of *Gzip-KNN* with that of deep learning techniques and the Google Bidirectional Encoder Representations from Transformers (BERT) model. Experimental results show that *Gzip-KNN* is competitive with deep learning methods and can outperform BERT in out-of-domain benchmark datasets, exemplifying its robustness in handling unseen data distributions.

In our work, we extend on the previous study by focusing on the detection of AI-generated text. More specifically, we evaluate the potential of *Gzip-KNN* for the detection of texts generated by ChatGPT. We frame the problem as a supervised classification task, where the objective is to learn a mapping between a representation of the text and a binary variable, which is 1 if the text is generated by ChatGPT, and 0 otherwise. Then, we compare the performance of *Gzip-KNN*, in terms of predictive power, training time, inference time and memory footprint, to that of other approaches. In particular, we consider both lightweight models, such as logistic regression and eXtreme Gradient Boosting (XGB), and more complex approaches, namely the Gated Recurrent Unit (GRU). We refrain from considering pre-trained models as our aim is to compare *Gzip-KNN* to low-resource approaches. To systematically discuss our findings, we pose the following research questions (RQs):

RQ1) To what extent can *Gzip-KNN* detect ChatGPT-generated text? Can *Gzip-KNN* outperform traditional ML-supervised approaches in terms of predictive performance?
RQ2) How does *Gzip-KNN* compare to other approaches in terms of training time and inference time?
RQ3) Can *Gzip-KNN* outperform traditional ML approaches in an out-of-domain context? And in a data-constrained and inference-time-constrained scenarios?

To address these RQs, we conduct two experiments. In the first, we analyze the trade-off between predictive performance and complexity of *Gzip-KNN* and the supervised learning approaches in an in-domain context. In the second, we perform evaluations considering constraints on available data and on inference time in an out-of-domain context. The experimental findings demonstrate that *Gzip-KNN* exhibits strong predictive performance, surpassing alternative methods, even when making predictions with a limited amount of data. However, it does come with the drawback of increased inference time, which restricts its suitability to situations where rapid decision-making is not critical. Nevertheless, the results also indicate that *Gzip-KNN* can deliver comparable performance to other methods when utilizing only a small fraction of the available data in an out-of-domain context.

The paper is organized as follows. Section 2 discusses related work. Section 3 describes the *Gzip-KNN* approach proposed in (Jiang et al., 2023). In Section 4 we describe the datasets and experimental setup, and in Section 5 we present and discuss experiment results. Finally, Section 6 concludes the paper.

2 Related Work

The detection of AI-generated text is currently receiving a great deal of attention, as the proliferation of AI-generated text, particularly from advanced language models such as ChatGPT, has led to growing concerns about the authenticity and reliability of textual content across diverse domains (Guo et al., 2023; Khalil and Er, 2023; Tian and Cui, 2023). Moreover, as AI-generated content becomes more prevalent in online interactions, news articles, customer support chats, and creative writing, the need to accurately distinguish between human-generated and AI-generated text has gained paramount significance.

The community has dedicated substantial efforts to developing sophisticated machine learning models capable of detecting AI-generated content (Pegoraro et al., 2023; Liu et al., 2023; Guo et al., 2023; He et al., 2023). In particular, zero-shot and one-shot techniques have gained attention as innovative approaches for text classification in general and for identifying AI-generated text in particular (Mitchell et al., 2023; Liu et al., 2023; Yan et al., 2018).

Other approaches rely on statistical properties (Gehrmann et al., 2019), linguistic features (Ma et al., 2023; Guo et al., 2023), information-theorical metrics such as entropy (Gehrmann et al., 2019) and perpexity (Tian and Cui, 2023; Guo et al., 2023), topological features (of attention maps generated by the transformer model)

(Kushnareva et al., 2021), Transformers (Bleumink and Shikhule, 2023), pretrained language models without (Bakhtin et al., 2019) or with fine-tuning (Solaiman et al., 2019; Mitrović et al., 2023; Chakraborty et al., 2023; Ippolito et al., 2019; Guo et al., 2023; Chiang et al., 2023; Ma et al., 2023), where in particular GPT-2 Output Detector is frequently used (Gao et al., 2023; Anderson et al., 2023).

While these proposed methods may achieve the desired predictive performance on in- and out-of-domain data, their demanding computational requirements and memory footprint is a substantial obstacle to their deployment. *Gzip-KNN* presents a lightweight and resource-efficient alternative to complex solutions for AI-generated text detection, leveraging an innovative combination of approaches (e.g., compression techniques) to perform text classification without prior training.

3 Gzip and K-Nearest Neighbors for Text Classification

Algorithm 1 Text Classification using Gzip-KNN

Sample t to be classified
Training dataset $\mathcal{D} = \{s_1, s_2, \ldots, s_n\}$
Number of nearest neighbors k
function CLASSIFY($t, \mathcal{D}, k$)
  Compress t using Gzip (denote as $gzip(t)$)
  for each sample s in $\mathcal{D}$ **do**
    Compress s using Gzip ($gzip(s)$)
    Compute Normalized Compression Distance (NCD) between $gzip(s)$ and $gzip(t)$
    Store NCD in a distance list
  end for
  Find the indices of the k smallest distances in the distance list
  Retrieve the corresponding k nearest neighbors' classes
  Count the occurrences of each class among the k neighbors
  Pick the majority class as the target label for t
end function

In this Section, we describe the various steps executed by the *Gzip-KNN* algorithm to classify a sample text *t*. The corresponding pseudocode is shown in Algorithm 1. The first step involves compressing *t* using the *Gzip* algorithm. Then, for each sample s in the training dataset, the text is similarly compressed using *Gzip*, and subsequently, the Normalized Compression Distance (NCD) between the compressed form of s and t is calculated (see Equation 1).

$$NCD(t,s) = \frac{C(st) - \min\{C(t), C(s)\}}{\max\{C(t), C(s)\}} \quad (1)$$

where st represents the concatenation of texts t and s, while $C(\cdot)$ is the length of a text compressed using *Gzip*.

The NCD serves as a measure that indicates the extent of information shared between two distinct texts. When two texts exhibit substantial shared content, their concatenation yields a more efficient compression outcome, resulting in a reduced NCD value. Therefore, since texts belonging to the same class typically share a greater degree of common attributes compared to texts from distinct classes, the NCD value can be leveraged in the task of text classification. Specifically, the *Gzip-KNN* algorithm uses the NCD distance computation as the basis for identifying the k-nearest neighbors of a reference text *t* within the training set. Finally, the target text *t* is classified based on the majority label among the selected k-nearest neighbors.

The absence of tunable parameters makes training lightweight and straightforward. However, it must be noted that classifying a text sample *t* requires repeating the concatenation between *t* and all the samples of the training set, which may result in a high inference time, especially with large datasets. In Section 5 we analyze the impact of the size of the training set on both predictive performance and inference time.

4 Dataset

We choose a labeled dataset consisting of human- (class 0) and ChatGPT-generated (class 1) responses to a set of queries. Specifically, ChatGPT answers were generated using GPT-3.5. These responses are provided in relation to a set of queries that encompass a wide range of open-ended questions. These questions were drawn from five diverse datasets, each contributing queries representative of a specific domain:

- *open_qa*: General queries on various topics sourced from the WikiQA dataset (Yang et al., 2015).

- *wiki_csai*: Queries related to specific concepts within the realm of information technology, gathered from Wikipedia (Guo et al., 2023).
- *finance*: Queries centered around finance-related subjects, obtained from the FiQA dataset (Maia et al., 2018).
- *medicine*: Queries focused on the field of medicine, collected from the Medical Dialog dataset (Zeng et al., 2020).
- *reddit_eli5*: Open-ended questions spanning various subjects, gathered from the ELI5 dataset (Fan et al., 2019).

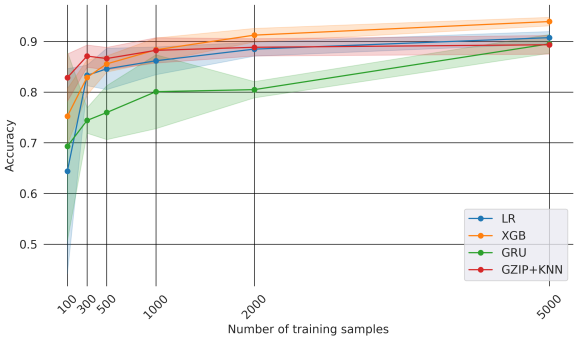
The human- and ChatGPT-generated responses are of similar length distribution (Guo et al., 2023).

5 Experimental Setup and Quantitative Evaluation

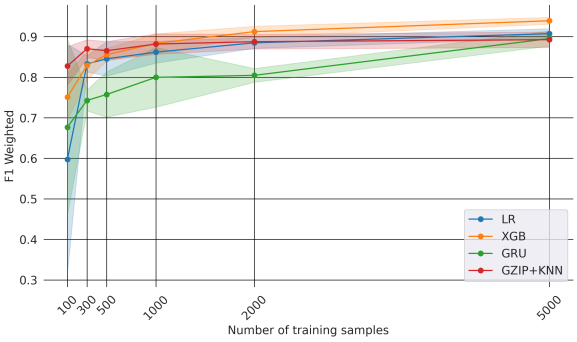
In this section, we present the results of our experiments, which aim to evaluate the *Gzip-KNN* algorithm for text classification along various dimensions. Specifically, in Section 5.1 we first evaluate the classification performance and the complexity of the approach, in terms of training time and inference time. Then, in Section 5.2, we assess the classification performance in an out-of-domain setting, where the algorithm is tested on datasets never seen during training.

5.1 Experiment 1: Performance vs. Complexity

Experimental Setup: We conduct this experiment while systematically varying the number of responses used during the training phase, all while ensuring a balanced distribution between the human-generated and ChatGPT-generated responses. Specifically, we consider a range of responses n = *100, 300, 500, 1000, 2000, 5000*, which are randomly selected from the dataset. We adopt a 5-fold cross-validation methodology for each value of n. During the testing phase, we evaluate the approach using a set of 10,000 responses, selected randomly from the dataset and equally split between the two labels. We also ensure the same testing sets are used to evaluate the different models for fold. The aim of this analysis is to comprehensively assess the model's performance across different training data volumes while maintaining a consistent test set.



(a) Classification Accuracy for varying training size



(b) F1-score for varying training size

Figure 1: Classification performance of the LR, XGB, GRU and Gzip-KNN models, for varying sizes of the training set

We compare *Gzip-KNN* to Logistic Regression (LR), eXtreme Gradient Boosting (XGB), and Gated Recurrent Unit (GRU) along two main dimensions, namely predictive performance and model complexity. To evaluate the former, we consider traditional classification metrics, such as *accuracy* and *F1-score*. To evaluate the latter, we consider the training time $t_{training}$ and the inference time $t_{inference}$.

Predictive Performance: Figure 1(a) shows the mean accuracy and standard deviation of the four models with respect to the number of training samples, ranging between 100 and 5000 training samples. When only 100 training samples are used, *Gzip-KNN* achieves an accuracy of 0.83, outperforming XGB (0.75), GRU (0.69), and LR (0.65). The significant gap in accuracy reveals *Gzip-KNN*'s capability in text classification, and particularly, in detecting ChatGPT-generated text, with very little training data. The accuracy of all models shows a general upward trend in performance as the training dataset size increases with *Gzip-KNN* outperforming the other approaches. However, it is worth noticing that the performance gap between *Gzip-*

KNN and the other approaches narrows as the training dataset size grows, up to a number of training samples equal to 1000. Specifically, for a number of training samples of 1000, *Gzip-KNN* shows an accuracy of 0.88, just slightly higher than that of XGB (0.87), LR (0.86), and GRU (0.81). The accuracy achieved by all models continues to increase as the training set size becomes larger, except for *Gzip-KNN*, which seems to saturate around an accuracy of 0.89, outperformed by XGB, LR, and GRU, which achieve an accuracy of 0.94, 0.91 and 0.9 using 5000 training samples, respectively.

Figure 1(b) shows the weighted F1-score and its standard deviation achieved by the different approaches with respect to the number of training samples, ranging between 100 and 5000. Results in terms of F1-score show a similar trend to that of accuracy. For a number of training samples less than 1000, *Gzip-KNN* outperforms other approaches, achieving an F1-score of around 0.88. This confirms *Gzip-KNN*'s intrinsic ability to distill and compress information effectively, even when the dataset is not exceedingly large, and that its architecture inherently adapts to the complexity of the data, discerning relevant features and connections without the need for a large number of examples. For a higher number of training samples, the F1-score of *Gzip-KNN* saturates around 0.89 while that of other approaches continues to show a slight increase as the size of the training dataset increases.

Overall, results show that the performance differences between *Gzip-KNN* and the other models, namely, XGB, LR, and GRU, tend to diminish as the training size increases, which could be due to more data being available for XGB, LR, and GRU, reducing overfitting and improving generalization, while *Gzip-KNN* does not further benefit from more training samples. In other words, the performance gains achieved by the *Gzip-KNN* seem to saturate beyond a certain point of dataset size. Unlike traditional methods that tend to improve as more data is fed into their training pipelines, *Gzip-KNN* appears to capitalize on a specific threshold of data sufficiency. This suggests that, for *Gzip-KNN*, the emphasis should be placed not solely on increasing the dataset size, but rather on refining the compression and distance calculation mechanisms. Further research could delve into optimizing the interplay between these two components to extract more nuanced information and potentially push the *Gzip-KNN*'s performance boundaries.

Model Complexity: We first examine the complexity of the considered approaches in terms of training time and inference time. Table 1 shows the training time (in seconds), averaged over 10 different evaluations, for the various models with respect to the number of training observations. Results show that the training times of the various models exhibit distinct trends as the number of training observations increases. The *Gzip-KNN*, which demonstrates exceptional efficiency, consistently yielding remarkably low training times across, shows a slightly increasing trend, ranging between 0.005 seconds for 100 training observations to 0.093 seconds for 2000 training observations. LR and XGB demonstrate linear increments in training time with the expansion of training data, reaching up to 0.274 seconds for LR and 0.810 seconds for XGB, at 2000 training observations. GRU, on the contrary, shows a nuanced pattern, with training times displaying some fluctuations without a clear trend, ranging between 8.7 and 13.3 seconds, on average. These results highlight the clear advantage *Gzip-KNN* has over other models in terms of training time, which suggests its potential utility for scenarios demanding swift model deployment.

We now focus on the inference time. Table 2 reports the variation in inference times for different models, across various sizes of the training set. Notably, the *Gzip-KNN* approach consistently exhibited relatively higher inference times compared to the other models for all sizes of the training set, ranging between 5.9 seconds (when 100 observations are used to perform the inference) to 115.6 seconds (when 2000 observations are used to perform the inference). On the contrary, the inference time for other approaches is significantly lower (i.e., fluctuating around 1 second), and not dependent on the size of the training set. This shows that the *Gzip-KNN* approach introduces an additional significant computational overhead with respect to other approaches, particularly when the number of observations used for inference is relatively large.

Gzip-KNN: Performance vs. Complexity. Focusing our attention on *Gzip-KNN* approach, a distinct trade-off emerges between predictive power and the time required for inference. As illustrated in Figure 1, *Gzip-KNN* achieves a relatively high predictive performance (0.82 of accuracy and 0.85 of F1-score) even with a small number of obser-

Table 1: Training Time Results

Num. Training Observations	Training Time (seconds)			
	GRU	GZIP+KNN	LR	XGB
100	10.101	0.005	0.017	0.093
300	9.497	0.014	0.040	0.168
500	8.701	0.023	0.054	0.230
1000	14.179	0.048	0.111	0.458
2000	13.346	0.093	0.275	0.811
5000	28.238	0.225	0.496	1.697

Table 2: Inference Time Results

Num. Training Observations	Inference Time (seconds)			
	GRU	GZIP+KNN	LR	XGB
100	1.033	5.944	0.072	0.096
300	0.922	17.332	0.074	0.095
500	0.781	28.848	0.076	0.095
1000	1.030	58.784	0.091	0.100
2000	0.884	115.589	0.083	0.099
5000	0.927	286.129	0.077	0.103

vations employed for inference (100 observations). This, however, corresponds to a relatively elevated inference time of 5 seconds (see Table 2). The predictive performance of *Gzip-KNN* can be further improved to reach 0.9 accuracy by employing a larger set of observations for inference (1000 samples). However, this incurs a substantial increase in inference time, culminating in an extended duration of 58.5 seconds. This suggests that the *Gzip-KNN* approach can have a robust predictive performance, particularly when dealing with a constrained dataset. Yet, its value is limited to scenarios where prediction accuracy takes precedence over rapid inference. In conclusion, while *Gzip-KNN* offers a powerful tool for predictive tasks, its optimal use hinges on aligning its strengths with the specific requirements of the given application context.

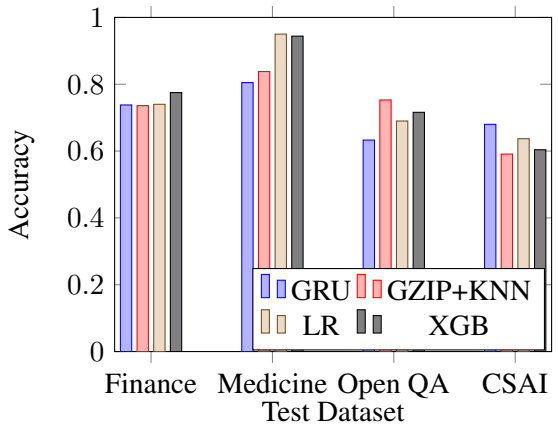
5.2 Experiment 2: Performance in Out-of-Domain Context

Experimental Setup: We now shift our attention to assessing the performance of the different methods in an out-of-domain context in different circumstances. Specifically, we perform two evaluations. In the first evaluation, we consider that a limited amount of data is available for training. The objective of this experiment is to quantify the capability of *Gzip-KNN* in detecting ChatGPT-generated text in an out-of-domain context and under the limitations of available data. We perform the training considering a part of the datasets, set at 1000 text samples, extracted from three specific contexts (e.g., from technology, finance and open QA datasets) with equal contribution and then perform the testing on a different dataset that corresponds to a different context (e.g., medicine). Note that while no formal training takes place for the *Gzip-KNN* approach, the inference still relies on the utilization of text samples, which are the training samples used to train the other ML models.

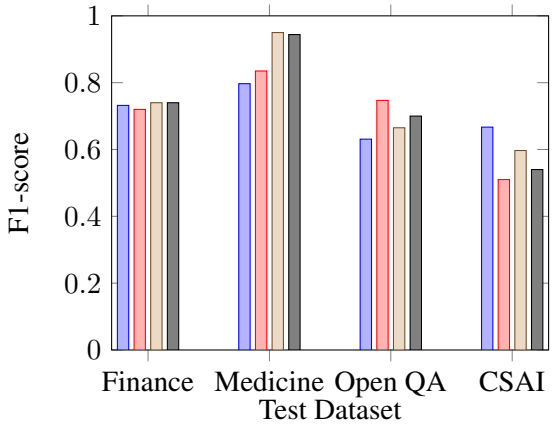
In the second evaluation, we introduce a constraint on inference time. To comply with the imposed inference time constraint, the size of the data used by *Gzip-KNN* at inference time must be restricted. Specifically, 600 samples are randomly taken from the training set, and used to perform the inference. On the contrary, for the other approaches, since using all data available for training does not heavily impact the inference time, we consider that all data available can be used for training. The objective of this experiment is to asses whether *Gzip-KNN* can outperform other models in an out-of-domain context even when a limit is imposed on inference time (and, therefore, on the amount of data that are required by *Gzip-KNN* to perform a text classification).

Out-of-Domain ChatGPT-generated Text Detection with Limited Data: Figures 2(a) and 2(b) show the accuracy and F1-score metrics, respectively, that are achieved by the four models when tested on the considered datasets. In general, XGB and LR tend to outperform other approaches, consistently achieving some of the highest performance levels across most datasets. For instance, in the Finance dataset, XGB achieves the best accuracy and F1-score (0.807 and 0.805, respectively) and ranks second in the Medicine and OpenQA datasets (e.g., its accuracy is 0.944 and 0.716, respectively). LR achieves the highest accuracy and F1-score in the Medicine dataset (0.95 for both metrics) and it ranks second in the Finance and CSAI datasets, with accuracy values of 0.74 and 0.69, respectively. *Gzip-KNN* generally achieves lower performance compared to alternative methods. However, it is important to note that *Gzip-KNN* reaches the best performance in the OpenQA dataset, surpassing alternative methods in both accuracy and F1-score, with values of 0.753 and 0.665, respectively. Additionally, in the Finance dataset, the performance of *Gzip-KNN* is only slightly lower than that of the alternatives (indeed, GRU, *Gzip-*

KNN, and LR all achieve an accuracy of around 0.74).



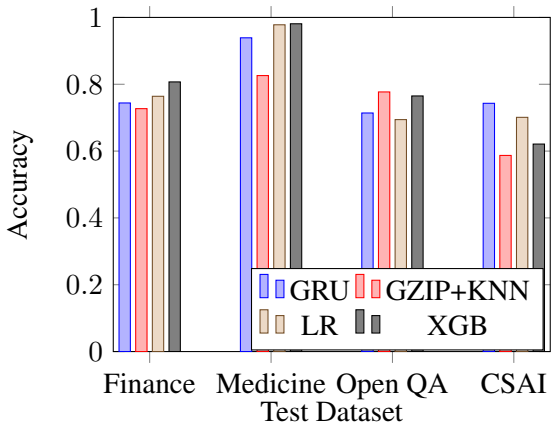
(a) Accuracy of Different Models on Various Datasets



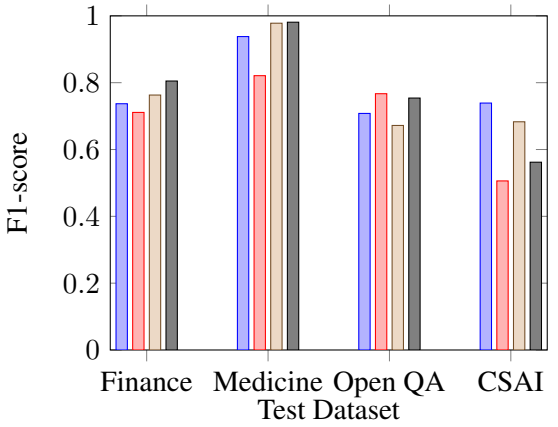
(b) F1-score of Different Models on Various Datasets

Figure 2: Comparison of accuracy and F1-score of different models for out-of-domain ChatGPT-generated text detection considering different test datasets.

Out-of-Domain ChatGPT-generated Text Detection under Constraint on Inference Time: Figures 3(a) and 3(b) show the accuracy and F1-score metrics of the four different approaches for each of the cases, respectively. Notably, *XGB* exhibits consistently high accuracy and F1-scores, specifically for Medicine and Finance (0.981 accuracy in both cases), outperforming other approaches in all cases except for when testing over the CSAI dataset. GRU also shows a similar performance outperforming other approaches for the case of testing over the CSAI dataset. *LR* demonstrates competitive performance outperforming GRU in some cases while *Gzip-KNN* achieves the highest accuracy on the Medicine dataset (0.826) and the lowest on the CSAI dataset (0.587). With respect to other approaches, *Gzip-KNN* achieved generally lower, yet comparable, accuracy and F1-scores, except when testing on Open QA, where it achieved the highest accuracy and F1-score. Note that *Gzip-KNN* uses only 600 text samples for this experiment, while other approaches utilize all available datasets. This shows that *Gzip-KNN* can achieve performance in out-of-domain ChatGPT-generated text detection when using a significantly small amount of data (a portion of the dataset) comparable to that of other approaches (in this case, LR, GRU, and XGB) when trained on the entire dataset.



(a) Accuracy of Different Models on Various Datasets



(b) F1-score of Different Models on Various Datasets

Figure 3: Comparison of accuracy and F1-score of different models for out-of-domain ChatGPT-generated text detection considering different test datasets.

6 Conclusion

In this work, we evaluate the effectiveness of a recently proposed algorithm, *Gzip-KNN*, in the task of detection of ChatGPT-generated text. The *Gzip-KNN* algorithm combines compression techniques with the k-nearest neighbors (KNN) algorithm for classification, resulting in a lightweight solution compared to traditional techniques used for text classification. Specifically, we compare this ap-

proach with LR, XGB, and GRU, in terms of classification performance and model complexity in various scenarios. Obtained results show that the *Gzip-KNN* algorithm outperforms the alternatives in terms of classification performance in situations where the training dataset is limited in the number of samples. However, such an advantage comes also with an increased inference time, which is significantly higher for *Gzip-KNN* than for the other approaches. Finally, we also evaluated the classification performance of the approaches in an out-of-domain setting, where the models are tested on a set never seen during training. These experiments have shown that *Gzip-KNN* can yield comparable classification performance to the other methods while only utilizing a significantly lower amount of training data.

References

Nash Anderson, Daniel L Belavy, Stephen M Perle, Sharief Hendricks, Luiz Hespanhol, Evert Verhagen, and Aamir R Memon. 2023. Ai did not write this manuscript, or did it? can we trick the ai text detector into generated texts? the potential future of chatgpt and ai in sports & exercise medicine manuscript generation.

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. *arXiv preprint arXiv:1906.03351*.

Arend Groot Bleumink and Aaron Shikhule. 2023. Keeping ai honest in education: Identifying gpt-generated text. *Edukado AI Research*, pages 1–5.

Souradip Chakraborty, Amrit Singh Bedi, Sicheng Zhu, Bang An, Dinesh Manocha, and Furong Huang. 2023. On the possibilities of ai-generated text detection. *arXiv preprint arXiv:2304.04736*.

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.

Francois Chollet. 2017. The limitations of deep learning. *Deep learning with Python*.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy. Association for Computational Linguistics.

Quentin Fournier, Gaétan Marceau Caron, and Daniel Aloise. 2023. A practical survey on faster and lighter transformers. *ACM Computing Surveys*, 55(14s):1–40.

Catherine A Gao, Frederick M Howard, Nikolay S Markov, Emma C Dyer, Siddhi Ramesh, Yuan Luo, and Alexander T Pearson. 2023. Comparing scientific abstracts generated by chatgpt to real abstracts with detectors and blinded human reviewers. *NPJ Digital Medicine*, 6(1):75.

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

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. 2023. How close is chatgpt to human experts? comparison corpus, evaluation, and detection. *arXiv preprint arXiv:2301.07597*.

Suchin Gururangan, Tam Dang, Dallas Card, and Noah A Smith. 2019. Variational pretraining for semi-supervised text classification. *arXiv preprint arXiv:1906.02242*.

Xinlei He, Xinyue Shen, Zeyuan Chen, Michael Backes, and Yang Zhang. 2023. Mgtbench: Benchmarking machine-generated text detection. *arXiv preprint arXiv:2303.14822*.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2019. Automatic detection of generated text is easiest when humans are fooled. *arXiv preprint arXiv:1911.00650*.

Zhiying Jiang, Matthew Yang, Mikhail Tsirlin, Raphael Tang, Yiqin Dai, and Jimmy Lin. 2023. "low-resource" text classification: A parameter-free classification method with compressors. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6810–6828.

Mohammad Khalil and Erkan Er. 2023. Will chatgpt get you caught? rethinking of plagiarism detection. *arXiv preprint arXiv:2302.04335*.

Laida Kushnareva, Daniil Cherniavskii, Vladislav Mikhailov, Ekaterina Artemova, Serguei Barannikov, Alexander Bernstein, Irina Piontkovskaya, Dmitri Piontkovski, and Evgeny Burnaev. 2021. Artificial text detection via examining the topology of attention maps. *arXiv preprint arXiv:2109.04825*.

Zhengliang Liu, Xiaowei Yu, Lu Zhang, Zihao Wu, Chao Cao, Haixing Dai, Lin Zhao, Wei Liu, Dinggang Shen, Quanzheng Li, et al. 2023. Deid-gpt: Zero-shot medical text de-identification by gpt-4. *arXiv preprint arXiv:2303.11032*.

Yongqiang Ma, Jiawei Liu, Fan Yi, Qikai Cheng, Yong Huang, Wei Lu, and Xiaozhong Liu. 2023. Ai vs. human – differentiation analysis of scientific content generation.

Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. 2018. Www'18 open challenge: Financial opinion mining and question answering. In *Companion Proceedings of the The Web Conference 2018*, WWW '18, page 1941–1942, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. 2023. Detectgpt: Zero-shot machine-generated text detection using probability curvature. *arXiv preprint arXiv:2301.11305*.

Sandra Mitrović, Davide Andreoletti, and Omran Ayoub. 2023. Chatgpt or human? detect and explain. explaining decisions of machine learning model for detecting short chatgpt-generated text. *arXiv preprint arXiv:2301.13852*.

Chongyu Pan, Jian Huang, Jianxing Gong, and Xingsheng Yuan. 2019. Few-shot transfer learning for text classification with lightweight word embedding based models. *IEEE Access*, 7:53296–53304.

Alessandro Pegoraro, Kavita Kumari, Hossein Fereidooni, and Ahmad-Reza Sadeghi. 2023. To chatgpt, or not to chatgpt: That is the question! *arXiv preprint arXiv:2304.01487*.

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

Neil C Thompson, Kristjan Greenewald, Keeheon Lee, and Gabriel F Manso. 2020. The computational limits of deep learning. *arXiv preprint arXiv:2007.05558*.

Edward Tian and Alexander Cui. 2023. Gptzero: Towards detection of ai-generated text using zero-shot and supervised methods.

Leiming Yan, Yuhui Zheng, and Jie Cao. 2018. Few-shot learning for short text classification. *Multimedia Tools and Applications*, 77:29799–29810.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. WikiQA: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018, Lisbon, Portugal. Association for Computational Linguistics.

Guangtao Zeng, Wenmian Yang, Zeqian Ju, Yue Yang, Sicheng Wang, Ruisi Zhang, Meng Zhou, Jiaqi Zeng, Xiangyu Dong, Ruoyu Zhang, Hongchao Fang, Penghui Zhu, Shu Chen, and Pengtao Xie. 2020. MedDialog: Large-scale medical dialogue datasets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9241–9250, Online. Association for Computational Linguistics.