David versus Goliath: Can Machine Learning Detect LLM-Generated Text? A Case Study in the Detection of Phishing Emails

Francesco Greco[1,*], Giuseppe Desolda[1], Andrea Esposito[1] and Alessandro Carelli[1]

[1]University of Bari Aldo Moro, Via E. Orabona 4, 70125 Bari, Italy

Abstract

Large Language Models (LLMs) offer numerous benefits, but they also pose threats, such as cybercriminals creating fake, convincing content such as phishing emails. LLMs are more convenient for criminals than handcrafting, making phishing campaigns more likely and more widespread in the future. To combat these attacks, detecting whether an email is generated by LLMs is critical. However, previous attempts have resulted in solutions that are uninterpretable and resource-intensive due to their complexity. This results in warning dialogs that do not adequately protect users. This work aims to address this problem using traditional, lightweight machine learning models that are easy to interpret and require fewer computational resources. This approach allows users to understand why an email is AI-generated, improving their decision-making in the case of phishing emails. This study has shown that logistic regression can achieve excellent performance in detecting emails generated by LLMs, while still providing the transparency needed to provide useful explanations to users.

Keywords

phishing detection, explanation, warning dialogs, machine learning, large language models

1. Introduction

In an era marked by the proliferation of digital communication channels, phishing attacks are a growing concern for individuals, enterprises, and organizations [1]. Phishing is a cyber-attack in which malicious users deceive to steal sensitive information, such as passwords, financial details, and personal data. In recent years, this attack escalated with the introduction of Large Language Models (LLMs) designed as a 'black hat' alternative to traditional GPT models, allowing hackers to automate phishing and other malicious cyber-attacks, operating without ethical limits or restrictions. Such LLMs are highly successful since they aid attackers in generating highly convincing, tailored, and contextually relevant text, making it even more challenging to distinguish between legitimate content and malicious phishing attempts.

State-of-the-art solutions for detecting phishing attacks relied upon rule-based systems, blacklists, machine learning, and heuristic analysis [2, 3]. Although these approaches have been somewhat effective in detecting phishing content, they struggle to keep up with the constant

ITASEC 2024: The Italian Conference on CyberSecurity, April 08–12, 2024, Salerno, Italy

*Corresponding author.

francesco.greco@uniba.it (F. Greco); giuseppe.desolda@uniba.it (G. Desolda); andrea.esposito@uniba.it (A. Esposito); a.carelli5@studenti.uniba.it (A. Carelli)

0000-0003-2730-7697 (F. Greco); 0000-0001-9894-2116 (G. Desolda); 0000-0002-9536-3087 (A. Esposito)

updates and evolutions of phishing attacks. More recently, LLMs have also been used to classify LLM-based attacks [4, 5, 6]. Despite the plethora of solutions to detect phishing content, this attack remains very effective [1]. This problem is strongly related to the role of the victim in this attack, which is often neglected in the design of defensive solutions. Indeed, when phishing defensive systems identify a threat with a probability lower than a certain threshold (e.g., below 95%), they leave the user with a choice of what to do by showing a warning dialog. Even if the models used to classify the content have high accuracy, non-technological aspects, such as human factors [7], can lead users to ignore warnings. One such issue is the *habituation effect* [8]: when a user is repeatedly exposed to the same visual stimulus, like a phishing warning, they may eventually start to ignore its recommendations. Warning messages often contain technical or general information that may be difficult for users to comprehend. Research has demonstrated the significance of creating polymorphic warning interfaces in the context of phishing, which are interfaces that alter their appearance and/or content each time they are displayed to the user to reduce the habituation effect [9]. The second issue pertains to the *absence of clear explanations*. The provision of specific explanations within warnings has been shown to support users in making informed choices and thus reduce the risk of falling victim to phishing [10, 11, 12]. The third problem is the *distance between the different research fields* that study this attack: AI investigates classification models that perform as well as possible by focusing on metrics such as precision and recall; on the other hand, HCI focuses on the design of warnings and understanding of human factors, neglecting how phishing detection models can consider such aspects, for example, how models can provide explanations and how they can generate polymorphic content.

In an attempt to fill part of the gap in the literature, this study investigates machine learning models capable of detecting human- or LLM-generated phishing emails. Specifically, the models we investigated in this research are conceived as *post-hoc models* to be used in conjunction with already existing phishing detection systems (e.g., Google Safe Browsing), to provide a more powerful explanation to victims. Indeed, if these post-hoc models can establish if the email was LLM-generated, users can be warned with a more appropriate explanation, on the assumption that explanation is crucial in defending users against this attack. The choice of "traditional" machine learning models over LLMs or novel larger neural networks is two-faced: larger neural networks and LLMs are black-box models, hampering their explainability that can only be approximated using post-hoc techniques; furthermore, larger models have a significant requirement of computational resources and have a non-negligible impact on the environment [13], making them a worse choice for improving an existing classifier over other *green* smaller models.

We benchmarked 8 different machine learning (ML) models (i.e., random forest, SVM, XGBoost, logistic regression, K-nearest neighbors, naïve Bayes, and neural network) selected by reviewing the literature on LLM-generated text detection [14, 15, 16, 17, 18, 19]. The ML models were trained on a dataset comprising human-generated phishing emails [20] and LLM-generated phishing emails, created using WormGPT LLM [21]. Additionally, we trained a neural network on human- and LLM-generated generic text, and we then applied transfer learning by training the models on our dataset. To empower the training process with these datasets, we meticulously examined existing literature to identify pertinent text features utilized for distinguishing LLM-generated text [22, 23, 24, 25, 26], as well as text generated by artificial intelligence (AI) in general

[27, 28, 29]. A comprehensive set of 30 textual features was defined and used for encoding the datasets before the training phase.

The highest accuracy was obtained by the neural network without transfer learning (99.78%), but good performances were obtained by SVM (99.20%) and logistic regression (99.03%). We also compared the ML models, considering their ability to provide local explanations, i.e., their ability to provide information on the feature(s) that mainly contributed to the classification of the phishing email. Naïve Bayes model and logistic regression excel in providing local explanations, whereas other models such as neural networks, due to their black-box nature, necessitate supplementation with post-hoc eXplainable Artificial Intelligence (XAI) techniques like LIME [30] and SHAP [31]. While SHAP and LIME enable the explanation of black-box models, they inherently offer an approximation of the true rationale behind classification decisions. Thus, a trade-off between accuracy and quality of the explanation must be considered when choosing the right model for this task. From our perspective, the optimal compromise lies in adopting logistic regression.

The paper is structured as follows. Section 2 reports the background and related work on phishing detection solutions, LLMs and their use as phishing powering tools, and research in detecting AI-generated textual content. Section 3 describes the pipeline we used to train and test different ML models and their comparison. In Section 4, we discuss future works and draw conclusions.

2. Background and Related Work

2.1. Phishing Detection

Phishing is a problematic threat, as it leverages human vulnerabilities to succeed [7]. Therefore, to effectively offer protection against phishing attacks, both technological and human defenses should be put in place. Automated phishing detection is one of the main techniques to mitigate the problem of phishing [32, 33], and it comprises all the techniques for automatically detecting phishing content such as emails or websites. Generally, there are two main approaches to protect users from phishing attacks with detection techniques: the phishing content can be filtered to not allow it to reach the user in the first place, or the user can be warned about the threat.

One of the most used techniques for filtering dangerous content is to block phishing websites according to their presence on blacklists [34]. This approach allows to have very high precision in the detection (low false positive rate) since blacklisted websites are almost certainly malicious. The downside is that it takes time for the blacklists to be updated, and, therefore, a lot of false negatives can still reach the user in the case of zero-day attacks [34]. On the other hand, detection methods based on artificial intelligence (AI) are capable of also blocking unseen attacks, substantially improving the recall in this task [2]. However, AI-based detectors are not 100% accurate [3] and can still produce false positives (i.e., genuine emails/websites classified as phishing), which can ultimately jeopardize user productivity. Therefore, automatic filtering is only applicable to methods that have a very low chance of producing false positives, such as blacklists.

To ensure that the user can decide about emails or websites for which the classification is

uncertain, a common approach consists of displaying a warning dialog that alerts the user about the possible danger [35]. This can be applied, for example, to emails or websites that have been classified by an AI detector as "phishing" with a certain probability (e.g., in the 70-95% range). Warnings can persuade the user to steer clear of suspicious content, but commonly employed warnings are flawed, as they often lack explanations [10]. The lack of explanation about the specific danger places the burden of locating phishing cues on the user, who is often not an expert and does not possess the knowledge to make an informed decision [7]. Moreover, the lack of explanations can demotivate the user in heeding the warning and can lower the trust in the system [36]. Another problem with traditional warnings is that they retain the same aspect, even under different circumstances: this can easily produce a habituation effect in the users, who are much more likely to ignore the warning [8]. To reduce the habituation effect, warnings should be polymorphic, i.e., change their aspect (color, shape, content, etc.) with each interaction.

2.2. Large Language Models and LLM-powered Phishing Tools

Large Language Models (LLMs) represent one of the biggest technological advances in the field of Natural Language Processing. Currently, most LLMs are based on the Transformer architecture [37]; their staggering performance on human-like tasks [38] is mainly due to their massive number of parameters and the vast amount of data on which they are trained, which confers them the capability to identify subtle patterns in linguistic data and have access to extensive knowledge on several domains. Some of the most relevant commercially available LLMs are OpenAI's ChatGPT [39] and all the GPT models [40], PaLM 2 [41] and Gemini [42] by Google, Claude 2 [43] by Anthropic, and Meta's Llama 2 [44].

Cybercriminals did not waste time finding malicious uses of LLMs. Indeed, AI's impressive capabilities in creating human-like text can help fraudsters generate phishing emails that are more effective at deceiving users; producing convincing messages using LLMs also requires much less time and effort than crafting emails manually. LLM-generated content appears to possess critical properties for successful phishing attacks like convincingness, consistency, personalization, and fluency [45]. In a study by Hazell [46], GPT-3.5 and GPT-4 were used to produce spear-phishing emails directed to 600 British Members of Parliament, including collecting publicly available information; results showed that LLMs could considerably facilitate the conduction and scaling of spear-phishing attacks. A study by Sha [47] showed that GPT-3-generated phishing emails were less effective overall than human-crafted phishing emails. However, Heiding et al. [4] demonstrated that GPT-4 can generate the most effective phishing attacks when humans refine the emails produced by the model. This work shows that phishing campaigns powered by advanced LLMs like GPT-4 would be extremely advantageous for criminals, even if conducted in a completely automatic manner.

2.3. Detecting LLM-Generated Text

A first step towards the mitigation of phishing campaigns powered by LLMs is to detect whether an email is LLM-generated. Various efforts have been made in the literature toward this research direction, even though detecting AI-generated text without knowing the method used for the

generation still remains very tricky. There are various types of detectors for AI-generated text [48], but the most investigated category includes language models that are fine-tuned for the task.

These detectors are binary classifiers trained to discriminate between AI and human-generated content [49]. In 2019, OpenAI published GPT-2PD [50], a model based on RoBERTa [51] for detecting content produced by GPT-2 1.5B with an accuracy of ~95%. OpenAI then published a model for detecting generic AI-written text [52], but shut it down briefly after, as it had a very low performance in terms of recall (26%); nonetheless, this model was even described as significantly more reliable than the old GPT-2 detector [53]. Zellers et al. [54] proposed Grover, a transformer-based model for news generation, which is also used to detect AI-generated text. Using Grover itself to discriminate texts generated by Grover was indeed the most effective approach (~90% detection accuracy). GLTR [55] is a detector that uses both BERT and GPT-2 117M for detecting AI-generated text and offering users visual support to assist them in forensic analysis; the model itself achieved an AUC of about ~0.86, and it resulted to be effective in improving the user's performance in detecting AI-generated text (from 54% to 72%). Adelani et al. [56] compared Grover [54], GTLR [55], and GPT-2PD [50] on the detection of product reviews generated by GPT-2 fine-tuned on Amazon product reviews; the GPT-2 detector was the best at discriminating text generated by the GPT-2 model.

Fagni et al. [57] fine-tuned a RoBERTa-based model to detect AI-generated tweets in a dataset of deepfake tweets, obtaining an F1-score=0.896, outperforming both traditional ML models (e.g., bag-of-words) and complex neural network models (e.g., RNN, CNN) by a large margin. Uchendu et al. [58] employed a RoBERTa-based approach, which outperformed baseline detectors in spotting news articles generated by several TGMs (F1-score between ~0.85 and ~0.92). Finally, Mitrovic et al. [59] fine-tuned a DistilBERT model and used it to detect ChatGPT-generated text, obtaining excellent performance in a standard setting (accuracy=0.98), and decent performance in an adversarial setting (accuracy=0.79). Moreover, SHAP (Shapley Additive exPlanations) [31] was used to provide local explanations for specific decisions in the form of highlighting input text tokens.

DetectGPT [49] pertains to a different category of detectors, as it is not fine-tuned on any data for detecting LLM-generated content; in fact, it is a *zero-shot* detector, which uses different statistical signatures of AI-generated content to perform the detection. DetectGPT achieved, on average, ~0.95 AUROC in detecting content that was generated by different LLMs, across different datasets.

Watermarking is yet another technique used for detecting LLM-generated text. These detectors embed imperceptible signals in the generated medium itself so that they can later be detected efficiently [60]. An example of such detectors was presented by Kirchenbauer et al. [61].

All the mentioned detectors have the problem of being vulnerable to *paraphrasing* attacks, since also a light paraphraser can severely affect the reliability of the models [62]. Krishna et al. [63] proposed a *retrieval-based* detector, which seems to partially mitigate this vulnerability. This approach searches a database containing sequences of text previously generated by an LLM to detect LLM-generated content. The proposed algorithm looks for sequences that match the input text within a certain threshold. The authors empirically tested the tool using a database of 15M generations from a fine-tuned T5-XXL model, finding that it was able to detect 80% to 97%

of paraphrased generations across different settings while only classifying 1% of human-written sequences as AI-generated.

Another more traditional approach regards applying machine learning techniques for detecting AI-generated text. This involves using linguistic features extracted from the text such as TF-IDF (Term Frequency – Inverse Document Frequency) and bag-of-words [64] features (e.g., [57]), but also features like readability and understandability indexes (e.g., [58]). Various works address the problem with traditional ML models, including Naïve Bayes [65], SVM [19], Random Forest [17], XGBoost [18], multi-layer perceptron [16], and K-Nearest Neighbors [14]. In May 2019, OpenAI released a simple detector based on logistic regression that uses TF-IDF unigram and bigram features [66] that was able to detect GPT-2-generated content with an accuracy between 74% and 97% [50].

However, the huge number of parameters in LLMs (and other larger neural network-based techniques) requires a vast usage of computational resources for both training and usage. As they become more and more complex and widespread, their energy consumption and, thus, their carbon footprint become non-negligible [67]. Green AI [13] is a new field investigating how AI can be more environmentally friendly and inclusive. Lightweight models, e.g., traditional machine learning models such as random forests or shallow neural networks, can, therefore, be considered "green models" as they are a much more sustainable choice in terms of energy consumption.

3. Detecting Phishing Attacks Generated by LLMs

As a small step towards a polymorphic explainable model for phishing detection, we focus on detecting the author (i.e., humans or LLMs) of phishing emails using *green* AI models. The following section delves into the machine learning aspects of our work, providing details into the generation of the dataset, the training procedure, and the final results, providing a comparison among all tested machine learning models.

3.1. Materials

An appropriate dataset is needed to train machine learning models to discriminate between human-generated phishing emails and LLM-generated ones. With this goal in mind, we accessed a curated collection of human-generated phishing emails [20], selecting the most recent 1000 emails from the "Nazario" and "Nigerian Fraud" collections. To complete the dataset, we generated 1000 additional emails using an LLM. We adopted WormGPT, a version of ChatGPT fine-tuned to comply with malicious requests [21]. To generate the emails, the following prompt was used:

> *Pretend to be a hacker planning a phishing campaign. Generate 5 very detailed phishing emails, about* [topic] *using Cialdini's principle of* [principle]. *You have to use fake American real names for the sender and recipient (example: John Smith). Invent a phishing link URL for each email (example: https://refund-claim-link.com).*

In the prompt, two variables have been introduced to increase the variability of the email content. The "topic" variable determines the main message of the phishing email. The topics

that were selected and used for the generation are common topics for phishing emails: (i) Urgent Account Verification, (ii) Financial Transaction, (iii) Prize or Lottery Winning, (iv) Fake Invoice or Payment Request, (v) Charity Scam, (vi) Account Security, (vii) Tax Refund Scam, (viii) Job Offer, (ix) Social Media Notification, (x) COVID-19 Related Scam, (xi) Law Breaking Activity, (xii) War-Related Aid, and (xiii) Other random topics. The "principle" variable, instead, refers to Cialdini's six principles of persuasion [68], typically used in phishing emails to persuade users to perform malicious and dangerous actions. The values used in the prompts for the Cialdini principles were: (i) Reciprocity, (ii) Consistency, (iii) Social Proof, (iv) Authority, (v) Liking, (vi) Scarcity, and (vii) No principle.

The final dataset instances are labeled as either *positive* for LLM-generated content or *negative* for human-generated content. The dataset of raw emails is publicly available in a Kaggle dataset[1]. Since, as it will be better described in Section 3.2, we focus on machine learning models, we further process the dataset to extract features for the training phase. Referring to the literature, we extracted a total of 30 features [29]. Details on the features are available in the appendix (Table 2).

3.2. Methods

The overarching goal of our efforts is to provide an explainable green model for the discrimination of human-generated and LLM-generated phishing emails. For this reason, we used smaller classical machine learning models based on features rather than LLMs. To choose the best models for this task, we first analyzed the available literature. Most of the similar works focus on the following models: random forests [17], Support Vector Machines (SVM) [19], XGBoosting [18], Logistic Regression [66], K-Nearest Neighbors (KNN) [14], Naïve Bayes [65], and Neural Network [16]. To further expand the models' list, we also pre-trained the Neural Network on a dataset of various (not necessarily in the phishing context) emails written by either humans or LLM, and then fine-tuned it using our dataset.

Although these models are not always fully explainable by default, they are smaller and require fewer resources than bigger neural networks or transformer-based models [13]. To ensure consistent results, all models were implemented using Python, and the training phase was executed on a single machine. Furthermore, all models underwent a hyper-parameter selection phase to maximize the performances of each model for their comparison. The final parameters are available in the appendix (Table 3).

3.3. Experimental Results

The training phase for each model was executed on a single machine powered by a 13^{th}-generation Intel i7 processor and equipped with 16 GB of RAM. For these experiments, the use of a GPU was not required. To evaluate the proposed methods, we employed a *stratified repeated 10-fold cross-validation*. In other words, each fold contained roughly the same amount of positive and negative instances. For the neural network, we used the binary-cross entropy loss function, defined as:

[1] https://www.kaggle.com/datasets/francescogreco97/human-llm-generated-phishing-legitimate-emails

$$H(y, p) = -\left(y \log p + (1 - y) \log(1 - p)\right) \quad (1)$$

where y is the ground truth label, while p is the model output for an individual observation. Cross-entropy was minimized using the Adam optimizer and a fixed learning rate (whose value was optimized in the hyper-parameter selection phase).

We computed the *accuracy* as the performance metric, defined as the proportion of correctly classified instances (both true positive and true negatives) in the selected sample. Table 1 shows the average results of the repeated stratified 10-fold cross-validation for each model. The distribution of the accuracy is better represented in Figure 1.

Table 1
Average accuracy throughout the repeated 10-fold cross-validation

	Random Forest	SVM	XGBoost	Logistic Regression	KNN	Naïve Bayes	NN (Transfer Learning)	Neural Network
Average	98.16%	99.20%	97.50%	99.03%	97.67%	94.10%	99.06%	99.78%
Standard Deviation	0.0103	0.0057	0.0108	0.0055	0.0105	0.0165	0.0062	0.0034

By analyzing the results reported in Table 1, we can see that the Neural Network model seems to be the best-performing model, although the gain in accuracy is only 0.58% over the second-best model, SVMs. Among the better-performing models is logistic regression, which has an accuracy of 99.03%.

To better analyze the differences in performances of the various models, we performed a paired t-test for each model pair. The statistical test aims to understand whether one can reject the null hypotheses of the differences in the means being due only to chance. If the p-value was found to be less than 0.05, we calculated the effect size using Cohen's d score [69]. Since this score ranges from 0 to 1, to facilitate its interpretation, we categorized the effect sizes into three distinct levels: *insignificant* for values below 0.2, *low* for values ranging from 0.2 to 0.5, *medium* for values between 0.51 and 0.8, and *high* for values between 0.81 and 1. To facilitate the analysis of all these comparisons, in Figure 2 we depicted a matrix where each cell reports the p-value resulting from the comparison between the model specified in the related column and the model specified in the related cell. Furthermore, each cell is color-coded to represent the Cohen's d level: orange for high, yellow for medium, and green for low levels of effect size, while the cell has no color in case of insignificant values. In Figure 2 we can see that almost all differences in model accuracies are statistically relevant, except for the difference between KNN and XGBoost and the one between Logistic Regression and Neural Network (with Transfer Learning).

While the models investigated in our study demonstrate high performance, comparable even to the less interpretable and less green LLMs [13], it remains paramount to provide users with explanations regarding the malicious nature of content to defend against phishing attacks. For these reasons, our study also focuses on informing users whether an email originates from an AI source or not, detailing which aspect or feature of the text triggered suspicion, leading the

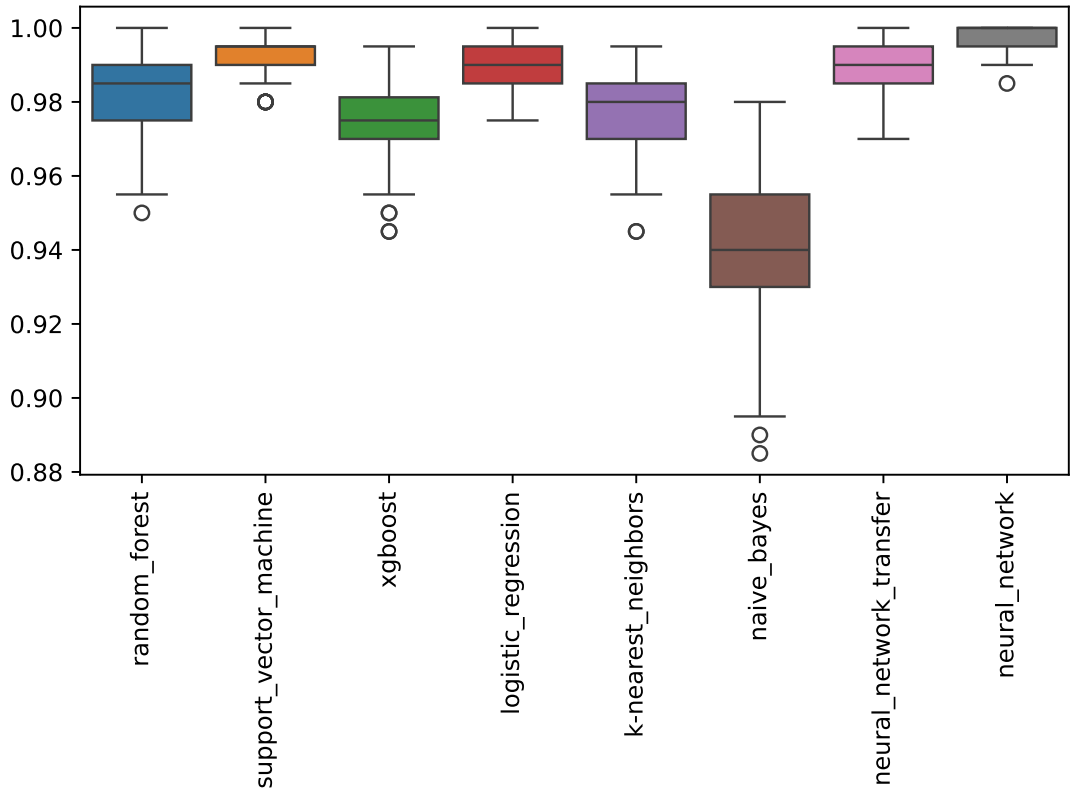


Figure 1: Box plot of the accuracy of each model for each cross-validation fold

defense system to classify it as human-written or AI-written. In line with other studies [12], the final goal is to warn users about phishing attacks with a warning dialog, as the one reported in Figure 3, which includes a message explaining that the email opened may have been generated by an AI.

Technically, this entails the ML model providing a *local explanation*, pinpointing the most influential feature among the 30 considered in the classification process for the classified email. Therefore, determining the best model for this task necessitates an analysis of each model's explanation capabilities. While models like logistic regression and K-nearest neighbors (KNN) are inherently explainable, the other models considered in this study require post-hoc models such as LIME or SHAP to provide explanations; however, in the case of the black-box models, the selected feature is an approximation of the one selected by the model, thus can be wrong and thus less effective in the explanation phase. Given logistic regression's innate ability to provide transparent explanations, together with its exceptional classification performance demonstrated in this study - virtually on a par with the best-performing neural networks - we argue that logistic regression is the most appropriate choice for detecting emails generated by LLMs, while providing essential explanations to users.

Figure 2: Heatmap showing the results of the paired t-test. Only statistically relevant differences are represented. The text in each cell is the computed p-value of the paired t-test. The color is used to represent the strength of the relationship, computed using Cohen's d.

4. Conclusion and Future Work

In this study, we analyzed different ML models for classifying emails as written by a human or using an LLM in the context of phishing. Detecting AI-generated emails can help mitigate the threat of phishing campaigns powered by LLMs, as these tools can produce convincing phishing emails in a fraction of the time it would otherwise take cybercriminals to create them manually. Therefore, we analyzed different ML models, which can be trained and used with less impact on the environment compared to LLMs.

Our experiments yielded interesting results: ML models were able to achieve accuracies above 90% in the task of classifying the author of the emails, in line with other works in the literature in other application domains (e.g., [54, 59, 49]). Considering the statistical relevance of the differences, the three best models (with an accuracy of over 99%) resulted to be Neural Networks, SVMs, and Logistic Regression (or Neural Networks with transfer learning). Although statistically relevant, the differences between the performances of the three models were only

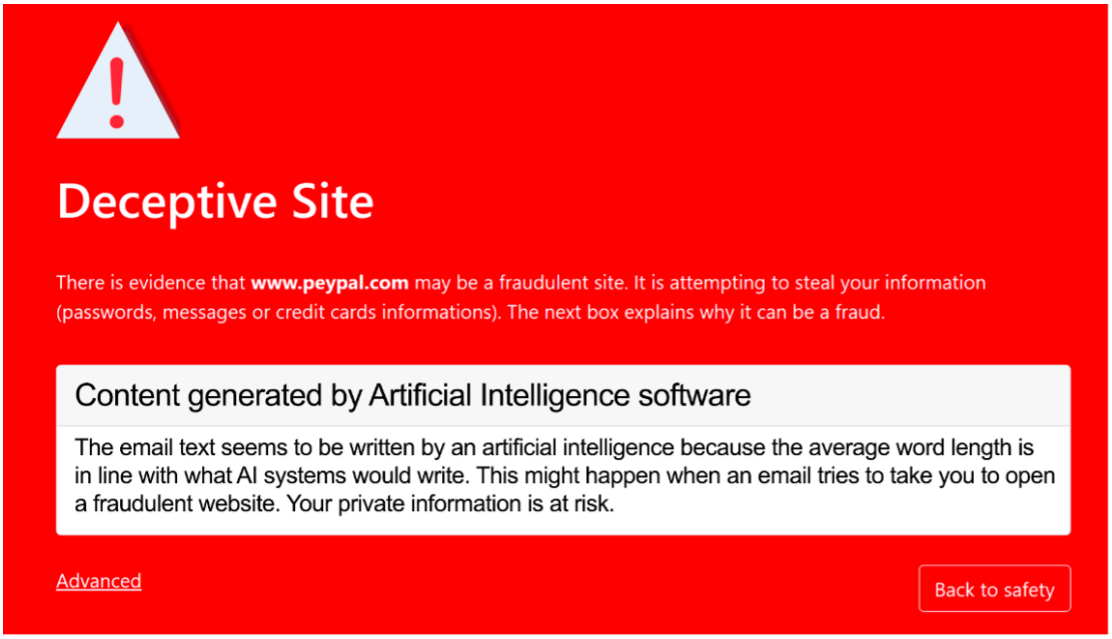


Figure 3: Example of warning dialogs used to warn users about a phishing attack; the warning includes an explanation that the email they opened may have been generated by an AI.

0.58% and 0.17%, respectively. However, neural networks are heavier to compute [13] and are difficult to explain [70]. Similarly, SVMs may also be difficult to interpret [71]. On the other hand, Logistic Regression is a simple white-box model and provides an easy way to interpret their results [72]. Having a transparent model allows us to interpret the model in terms of which features were more or less important in classifying a particular email as LLM-generated or not. This allows not only to use warning dialogs to warn the user when an email is classified as generated by an LLM, but also to provide an explanation. Explanations have the advantage of increasing the user's motivation to heed the warning dialog and their trust in the system [36]. Furthermore, using warning dialogs with explanations that change depending on the specific context, enhances the effectiveness of the warnings, as they reduce the user's habituation to seeing the same warning under different circumstances [73, 9]. However, to obtain warnings with these benefits, users must first understand the reported explanations [7]; therefore, if the explanations are based on reporting which features were most relevant in the ML model's decision, we must be able to effectively describe to the user what those features are. This means that not every feature of our feature set is adequate for constituting a good explanation for a naive user, as it may be overly technical.

Several future works are planned to extend and improve this research. First, we want to explore multi-class models that can detect phishing emails in general and determine whether the text is human-generated or not; unlike the post-hoc approaches proposed in this paper, multi-class models can be used as a stand-alone solution, useful in a scenario where post-hoc is not sufficient. Second, a user study is needed to determine which of the 30 features identified in our research can be explained to users, even without technical knowledge. Third, we aim

to benchmark our ML models in an adversarial setting, i.e., using paraphrasing attacks that introduce slight modifications in the LLM-generated emails, as even a light paraphraser can drastically decrease the effectiveness of detector tools [62]. Furthermore, it is also possible to extend the dataset, understanding whether the additional features, alongside others, can be used to detect phishing emails and their author using green and explainable machine learning models. Future studies may investigate if the slight loss of accuracy of simpler white-box models impacts the usefulness of the classifier through user studies and the inclusion of additional AI metrics (e.g., F1-score, precision, and recall). Finally, end-user development techniques will be explored to support the adaptation of the AI model and user interface to different contexts, with the aim of making the overall solution more tailored to specific needs [74, 75, 76].

Acknowledgments

This work is partially supported by the Italian Ministry of University and Research (MUR) under grant PRIN 2022 PNRR "DAMOCLES: Detection And Mitigation Of Cyber attacks that exploit human vuLnerabilitiES" – CUP: H53D23008140001.

This work is partially supported by the co-funding of the European Union - Next Generation EU: NRRP Initiative, Mission 4, Component 2, Investment 1.3 – Partnerships extended to universities, research centres, companies and research D.D. MUR n. 341 del 5.03.2022 – Next Generation EU (PE0000014 – "Security and Rights In the CyberSpace – SERICS" - CUP: H93C22000620001).

The research of Francesco Greco is funded by a PhD fellowship within the framework of the Italian "D.M. n. 352, April 9, 2022" - under the National Recovery and Resilience Plan, Mission 4, Component 2, Investment 3.3 - PhD Project "Investigating XAI techniques to help user defend from phishing attacks", co-supported by "Auriga S.p.A." (CUP H91I22000410007).

The research of Andrea Esposito is funded by a Ph.D. fellowship within the framework of the Italian "D.M. n. 352, April 9, 2022" - under the National Recovery and Resilience Plan, Mission 4, Component 2, Investment 3.3 - Ph.D. Project "Human-Centered Artificial Intelligence (HCAI) techniques for supporting end users interacting with AI systems", co-supported by "Eusoft S.r.l." (CUP H91I22000410007).

References

[1] IBM, Security x-force threat intelligence index, 2023. URL: https://www.ibm.com/reports/threat-intelligence.

[2] A. Almomani, B. B. Gupta, S. Atawneh, A. Meulenberg, E. Almomani, A survey of phishing email filtering techniques, IEEE Communications Surveys & Tutorials 15 (2013) 2070–2090. URL: https://ieeexplore.ieee.org/document/6489877. doi:10.1109/SURV.2013.030713.00020.

[3] M. Khonji, Y. Iraqi, A. Jones, Phishing detection: A literature survey, IEEE Communications Surveys & Tutorials 15 (2013) 2091–2121. URL: https://ieeexplore.ieee.org/document/6497928. doi:10.1109/SURV.2013.032213.00009.

[4] F. Heiding, B. Schneier, A. Vishwanath, J. Bernstein, P. S. Park, Devising and detecting phishing: Large language models vs. smaller human models, 2023. URL: https://doi.org/10.48550/arXiv.2308.12287. doi:arXiv:2308.12287.

[5] T. Koide, N. Fukushi, H. Nakano, D. Chiba, Detecting phishing sites using chatgpt, 2023. URL: https://arxiv.org/abs/2306.05816. doi:arXiv:2306.05816.

[6] M. Labonne, S. Moran, Spam-t5: Benchmarking large language models for few-shot email spam detection, 2023. URL: http://arxiv.org/abs/2304.01238. doi:arXiv:2304.01238.

[7] G. Desolda, L. S. Ferro, A. Marrella, T. Catarci, M. F. Costabile, Human factors in phishing attacks: A systematic literature review, 2021. URL: https://doi.org/10.1145/3469886. doi:10.1145/3469886.

[8] S. Kim, M. S. Wogalter, Habituation, dishabituation, and recovery effects in visual warnings, Human Factors and Ergonomics Society Annual Meeting 53 (2009) 1612–1616. URL: https://journals.sagepub.com/doi/abs/10.1177/154193120905302015. doi:10.1177/154193120905302015.

[9] B. B. Anderson, C. B. Kirwan, J. L. Jenkins, D. Eargle, S. Howard, A. Vance, How polymorphic warnings reduce habituation in the brain: Insights from an fmri study, in: ACM Conference on Human Factors in Computing Systems, ACM, Seoul, Republic of Korea, 2015, pp. 2883–2892. URL: https://doi.org/10.1145/2702123.2702322. doi:10.1145/2702123.2702322.

[10] C. Bravo-Lillo, L. F. Cranor, J. Downs, S. Komanduri, M. Sleeper, Improving computer security dialogs, in: International Conference on Human-Computer Interaction, volume LNCS, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 18–35. URL: https://dl.acm.org/doi/10.5555/2042283.2042286.

[11] P. Buono, G. Desolda, F. Greco, A. Piccinno, Let warnings interrupt the interaction and explain: designing and evaluating phishing email warnings, in: CHI Conference on Human Factors in Computing Systems, volume EA, ACM, Hamburg Germany, 2023, pp. 1–6. URL: https://dl.acm.org/doi/abs/10.1145/3544549.3585802. doi:10.1145/3469886.

[12] G. Desolda, J. Aneke, C. Ardito, R. Lanzilotti, M. F. Costabile, Explanations in warning dialogs to help users defend against phishing attacks, 2023. URL: https://www.sciencedirect.com/science/article/pii/S1071581923000654. doi:https://doi.org/10.1016/j.ijhcs.2023.103056.

[13] R. Schwartz, J. Dodge, N. A. Smith, O. Etzioni, Green ai, Communications of the ACM 63 (2020) 54–63. URL: https://doi.org/10.1145/3381831. doi:10.1145/3381831.

[14] H. Alshaher, J. Xu, A new term weight scheme and ensemble technique for authorship identification, in: International Conference on Compute and Data Analysis, ACM, Silicon Valley, CA, USA, 2020, pp. 123–130. URL: https://doi.org/10.1145/3388142.3388159. doi:10.1145/3388142.3388159.

[15] R. E. Roxas (Ed.), Stylometric Studies based on Tone and Word Length Motifs, Pacific Asia Conference on Language, Information and Computation, The National University (Phillippines), 2017. URL: https://aclanthology.org/Y17-1011.

[16] P. Sarzaeim, A. Doshi, Q. Mahmoud, A framework for detecting ai-generated text in research publications, in: International Conference on Advanced Technologies, volume 11, Istanbul, Turkey, 2023, pp. 121–127. URL: https://proceedings.icatsconf.org/conf/index.php/ICAT/article/view/36. doi:10.58190/icat.2023.28.

[17] A. Sharma, A. Nandan, R. Ralhan, An investigation of supervised learning methods for authorship attribution in short hinglish texts using char & word n-grams, 2018. URL: http://arxiv.org/abs/1812.10281. doi:arXiv:1812.10281.

[18] R. Shijaku, E. Canhasi, Chatgpt generated text detection, 2023. URL: http://dx.doi.org/10.13140/RG.2.2.21317.52960. doi:10.13140/RG.2.2.21317.52960.

[19] T. Solorio, S. Pillay, S. Raghavan, M. Montes y Gómez, Modality specific meta features for authorship attribution in web forum posts, in: International Joint Conference on Natural Language Processing, Asian Federation of Natural Language Processing, Chiang Mai, Thailand, 2011, pp. 156–164. URL: https://aclanthology.org/I11-1018.

[20] Anonymous, Phishing email curated dataset, 2023. URL: https://zenodo.org/records/8339691. doi:10.5281/zenodo.8339691.

[21] Forsasuke, Wormgpt, 2023. URL: https://flowgpt.com/p/wormgpt-6.

[22] L. Fröhling, A. Zubiaga, Feature-based detection of automated language models: tackling gpt-2, gpt-3 and grover, PeerJ Computer Science 7 (2021) 23. URL: https://doi.org/10.7717/peerj-cs.443. doi:10.7717/peerj-cs.443.

[23] P. Jwalapuram, S. Joty, X. Lin, Rethinking self-supervision objectives for generalizable coherence modeling, in: Annual Meeting of the Association for Computational Linguistics, volume 1, Association for Computational Linguistics, Dublin, Ireland, 2022, pp. 6044–6059. URL: https://doi.org/10.18653/v1/2022.acl-long.418. doi:10.18653/v1/2022.acl-long.418.

[24] Y. Ma, J. Liu, F. Yi, Q. Cheng, Y. Huang, W. Lu, X. Liu, Ai vs. human – differentiation analysis of scientific content generation, 2023. URL: http://arxiv.org/abs/2301.10416. doi:arXiv:2301.10416.

[25] A. Muñoz-Ortiz, C. Gómez-Rodríguez, D. Vilares, Contrasting linguistic patterns in human and llm-generated text, 2023. URL: http://arxiv.org/abs/2308.09067. doi:arXiv:2308.09067.

[26] T. T. Nguyen, A. Hatua, A. H. Sung, How to detect ai-generated texts?, in: Annual Ubiquitous Computing, Electronics & Mobile Communication Conference, IEEE, New York, USA, 2023, pp. 464–471. URL: https://ieeexplore.ieee.org/document/10316132. doi:10.1109/UEMCON59035.2023.10316132.

[27] R. Barzilay, M. Lapata, Modeling local coherence: An entity-based approach, Computational Linguistics 34 (2008) 1–34. URL: https://doi.org/10.1162/coli.2008.34.1.1. doi:10.1162/coli.2008.34.1.1.

[28] D. Kosmajac, V. Keselj, Twitter bot detection using diversity measures, in: International Conference on Natural Language and Speech Processing, Association for Computational Linguistics, Trento, Italy, 2019, pp. 1–8. URL: https://aclanthology.org/W19-7401.

[29] S. T. Piantadosi, Zipf's word frequency law in natural language: A critical review and future directions, Psychonomic Bulletin & Review 21 (2014) 1112–1130. URL: https://doi.org/10.3758/s13423-014-0585-6. doi:10.3758/s13423-014-0585-6.

[30] M. T. Ribeiro, S. Singh, C. Guestrin, "why should i trust you?": Explaining the predictions of any classifier, 2016. URL: http://arxiv.org/abs/1602.04938. doi:arXiv:1602.04938.

[31] S. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, 2017. URL: http://arxiv.org/abs/1705.07874. doi:arXiv:1705.07874v2.

[32] P. Kumaraguru, S. Sheng, A. Acquisti, L. F. Cranor, J. Hong, Teaching johnny not to

fall for phish, ACM Transactions on Internet Technology 10 (2010) 1–31. URL: https://doi.org/10.1145/1754393.1754396. doi:10.1145/1754393.1754396.
[33] G. Varshney, M. Misra, P. K. Atrey, A survey and classification of web phishing detection schemes, Security and Communication Networks 9 (2016) 6266–6284. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/sec.1674. doi:10.1002/sec.1674.
[34] S. Sheng, B. Wardman, G. Warner, L. Cranor, J. Hong, C. Zhang, An empirical analysis of phishing blacklists, in: International Conference on Email and Anti-Spam, Mountain View, California, USA, 2009. URL: https://kilthub.cmu.edu/articles/journal_contribution/An_Empirical_Analysis_of_Phishing_Blacklists/6469805/1. doi:10.1184/R1/6469805.v1.
[35] J. Petelka, Y. Zou, F. Schaub, Put your warning where your link is: Improving and evaluating email phishing warnings, in: CHI Conference on Human Factors in Computing Systems, ACM, Glasgow Scotland UK, 2019, pp. 1–15. URL: https://doi.org/10.1145/3290605.3300748. doi:10.1145/3290605.3300748.
[36] G. Vilone, L. Longo, Notions of explainability and evaluation approaches for explainable artificial intelligence, Information Fusion 76 (2021) 89–106. URL: https://doi.org/10.1016/j.inffus.2021.05.009. doi:10.1016/j.inffus.2021.05.009.
[37] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, 2017. URL: https://doi.org/10.48550/arXiv.1706.03762. doi:10.48550/arXiv.1706.03762.
[38] HuggingFace, Open llm leaderboard, 2024. URL: https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard.
[39] OpenAI, Introducing chatgpt, 2022. URL: https://openai.com/blog/chatgpt.
[40] OpenAI, Gpt-4 and gpt-4 turbo, 2023. URL: https://platform.openai.com/docs/models/gpt-4-and-gpt-4-turbo.
[41] Z. Ghahramani, Introducing palm 2 (2023). URL: https://blog.google/technology/ai/google-palm-2-ai-large-language-model/.
[42] G. DeepMind, Gemini, 2024. URL: https://deepmind.google/technologies/gemini.
[43] Anthropic, Claude 2 (2023). URL: https://www.anthropic.com/news/claude-2.
[44] Meta, Llama 2, 2023. URL: https://llama.meta.com/.
[45] D. Kang, X. Li, I. Stoica, C. Guestrin, M. Zaharia, T. Hashimoto, Exploiting programmatic behavior of llms: Dual-use through standard security attacks, 2023. URL: https://arxiv.org/abs/2302.05733. doi:arXiv:2302.05733.
[46] J. Hazell, Spear phishing with large language models, 2023. URL: https://arxiv.org/abs/2305.06972. doi:arXiv:2305.06972.
[47] How well does GPT phish people? An investigation involving cognitive biases and feedback, IEEE, 2023. URL: https://ieeexplore.ieee.org/document/10190709. doi:10.1109/EuroSPW59978.2023.00055.
[48] C. Barrett, B. Boyd, E. Bursztein, N. Carlini, B. Chen, J. Choi, A. R. Chowdhury, M. Christodorescu, A. Datta, S. Feizi, K. Fisher, T. Hashimoto, D. Hendrycks, S. Jha, D. Kang, F. Kerschbaum, E. Mitchell, J. Mitchell, Z. Ramzan, K. Shams, D. Song, A. Taly, D. Yang, Identifying and mitigating the security risks of generative ai, Foundations and Trends®in Privacy and Security 6 (2023) 1–52. URL: http://dx.doi.org/10.1561/3300000041. doi:10.1561/3300000041.
[49] E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, C. Finn, Detectgpt: Zero-shot machine-

generated text detection using probability curvature, 2023. URL: http://arxiv.org/abs/2301.11305. doi:arXiv:2301.11305.

[50] I. Solaiman, M. Brundage, J. Clark, A. Askell, A. Herbert-Voss, J. Wu, A. Radford, G. Krueger, J. W. Kim, S. Kreps, M. McCain, A. Newhouse, J. Blazakis, K. McGuffie, J. Wang, Release strategies and the social impacts of language models, 2019. URL: http://arxiv.org/abs/1908.09203. doi:arXiv:1908.09203.

[51] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, Roberta: A robustly optimized bert pretraining approach, 2019. URL: http://arxiv.org/abs/1907.11692. doi:arXiv:1907.11692.

[52] OpenAI, New ai classifier for indicating ai-written text, 2023. URL: https://openai.com/blog/new-ai-classifier-for-indicating-ai-written-text.

[53] OpenAI, Gpt-2 output detector, 2019. URL: https://github.com/openai/gpt-2-output-dataset/tree/master/detector.

[54] R. Zellers, A. Holtzman, H. Rashkin, Y. Bisk, A. Farhadi, F. Roesner, Y. Choi, Defending against neural fake news, 2020. URL: http://arxiv.org/abs/1905.12616. doi:arXiv:1905.12616v3.

[55] S. Gehrmann, H. Strobelt, A. M. Rush, Gltr: Statistical detection and visualization of generated text, 2019. URL: http://arxiv.org/abs/1906.04043. doi:arXiv:1906.04043.

[56] D. I. Adelani, H. Mai, F. Fang, H. H. Nguyen, J. Yamagishi, I. Echizen, Generating sentiment-preserving fake online reviews using neural language models and their human- and machine-based detection, 2019. URL: http://arxiv.org/abs/1907.09177. doi:arXiv:1907.09177.

[57] T. Fagni, F. Falchi, M. Gambini, A. Martella, M. Tesconi, Tweepfake: About detecting deepfake tweets, PLOS ONE 16 (2021) e0251415. URL: https://doi.org/10.1371/journal.pone.0251415. doi:10.1371/journal.pone.0251415.

[58] A. Uchendu, T. Le, K. Shu, D. Lee, Authorship attribution for neural text generation, in: Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Online, 2020, pp. 8384–8395. URL: https://aclanthology.org/2020.emnlp-main.673. doi:10.18653/v1/2020.emnlp-main.673.

[59] S. Mitrović, D. Andreoletti, O. Ayoub, Chatgpt or human? detect and explain. explaining decisions of machine learning model for detecting short chatgpt-generated text, 2023. URL: http://arxiv.org/abs/2301.13852. doi:arXiv:2301.13852.

[60] I. S. Moskowitz (Ed.), Natural Language Watermarking: Design, Analysis, and a Proof-of-Concept Implementation, volume LNCS, volume 2137 of *Information Hiding*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2001. URL: https://link.springer.com/chapter/10.1007/3-540-45496-9_14.

[61] J. Kirchenbauer, J. Geiping, Y. Wen, J. Katz, I. Miers, T. Goldstein, A watermark for large language models, 2023. URL: http://arxiv.org/abs/2301.10226. doi:arXiv:2301.10226v3.

[62] V. S. Sadasivan, A. Kumar, S. Balasubramanian, W. Wang, S. Feizi, Can ai-generated text be reliably detected?, 2023. URL: http://arxiv.org/abs/2303.11156. doi:arXiv:2303.11156v2.

[63] K. Krishna, Y. Song, M. Karpinska, J. Wieting, M. Iyyer, Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense, 2023. URL: http://arxiv.org/abs/2303.13408. doi:arXiv:2303.13408.

[64] F. Sebastiani, Machine learning in automated text categorization, ACM Computing

Surveys 34 (2002) 1–47. URL: https://doi.org/10.1145/505282.505283. doi:10.1145/505282.505283.

[65] F. Howedi, M. Masnizah, Text classification for authorship attribution using naive bayes classifier with limited training data, Computer Engineering and Intelligent Systems 5 (2014). doi:https://iiste.org/Journals/index.php/CEIS/article/view/12132/12484.

[66] OpenAI, Logistic regression gpt-2 detector, 2019. URL: https://github.com/openai/gpt-2-output-dataset/blob/master/baseline.py.

[67] R. Verdecchia, J. Sallou, L. Cruz, A systematic review of green ai, WIREs Data Mining and Knowledge Discovery 13 (2023) 26. URL: https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/widm.1507. doi:10.1002/widm.1507.

[68] R. B. Cialdini, Influence: The Psychology of Persuasion, Collins Business Essentials, revised ed., Harper Collins, 2009.

[69] J. Cohen, Statistical Power Analysis for the Behavioral Sciences, 2nd edition ed., Routledge, New York, USA, 1988. URL: https://doi.org/10.4324/9780203771587. doi:10.4324/9780203771587.

[70] O. Loyola-González, Black-box vs. white-box: Understanding their advantages and weaknesses from a practical point of view, IEEE Access 7 (2019) 154096–154113. URL: https://ieeexplore.ieee.org/document/8882211. doi:10.1109/ACCESS.2019.2949286.

[71] A. Navia-Vázquez, E. Parrado-Hernández, Support vector machine interpretation, Neurocomputing 69 (2006) 1754–1759. URL: https://www.sciencedirect.com/science/article/pii/S0925231205004480. doi:10.1016/j.neucom.2005.12.118.

[72] S. Meacham, G. Isaac, D. Nauck, B. Virginas, Towards explainable ai: Design and development for explanation of machine learning predictions for a patient readmittance medical application, in: K. Arai, R. Bhatia, S. Kapoor (Eds.), Intelligent Computing, volume 997, Springer, Cham, London, UK, 2019, pp. 939–955. URL: https://doi.org/10.1007/978-3-030-22871-2_67. doi:10.1007/978-3-030-22871-2{_}67.

[73] F. Greco, G. Desolda, A. Esposito, Explaining phishing attacks: An xai approach to enhance user awareness and trust, in: F. Buccafurri, E. Ferrari, G. Lax (Eds.), The Italian Conference on CyberSecurity, volume 3488, CEUR-WS, Bari, Italy, 2023. URL: https://ceur-ws.org/Vol-3488/paper22.pdf.

[74] C. Ardito, P. Bottoni, M. F. Costabile, G. Desolda, M. Matera, A. Piccinno, M. Picozzi, Enabling end users to create, annotate and share personal information spaces, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 7897 LNCS (2013) 40 – 55. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84884378360&doi=10.1007%2f978-3-642-38706-7_5&partnerID=40&md5=ac9ba219ee101062200d61f268479daa. doi:10.1007/978-3-642-38706-7_5.

[75] G. Desolda, Enhancing workspace composition by exploiting linked open data as a polymorphic data source, Smart Innovation, Systems and Technologies 40 (2015) 97 – 108. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84947913933&doi=10.1007%2f978-3-319-19830-9_9&partnerID=40&md5=2e4d49da34406b062da3f5f310e3b922. doi:10.1007/978-3-319-19830-9_9.

[76] C. Ardito, M. F. Costabile, G. Desolda, M. Latzina, M. Matera, Making mashups actionable

through elastic design principles, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 9083 (2015) 236 – 241. doi:10.1007/978-3-319-18425-8_22.

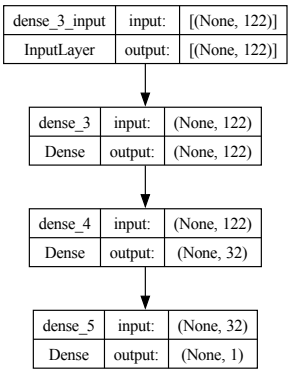
[77] G. Jawahar, M. Abdul-Mageed, L. Lakshmanan, V. S., Automatic detection of machine generated text: A critical survey, in: International Conference on Computational Linguistics, International Committee on Computational Linguistics, Barcelona, Spain (Online), 2020, pp. 2296–2309. URL: https://aclanthology.org/2020.coling-main.208https://doi.org/10.18653/v1/2020.coling-main.208. doi:10.18653/v1/2020.coling-main.208.

A. Appendix

Table 2
List of features

Feature name	Reference paper(s)
average_word_length	[27, 25]
pos_tag_frequency	[27, 77, 25]
uppercase_frequency	[27, 25]
average_sentence_length	[27, 77]
function_words_frequency	[27]
flesch_reading_ease	[27, 25]
type_token_ratio	[77]
dependency_types	[77]
emotions	[77]
named_entity_count	[28, 25]
common_words	[28]
stop_words	[28]
bigram	[28]
trigram	[28]
lack_of_purpose	[28]
word_distribution_zipf_law_slope	[26]
word_distribution_zipf_law_r_squared	[26]
word_distribution_zipf_law_cost	[26]
consistency_phrasal_verbs	[26]
text_diversity_yulek	[29]
text_diversity_simpsond	[29]
text_diversity_honorer	[29]
text_diversity_sichels	[29]
coherence_1	[23]
coherence_2	[27, 28]
constituent_lengths	[77]
constituent_types	[77]
coreference_resolution	[26]
lexical_diversity	[28]

Table 3
Models' parameters and architectures, reported as provided by SciKit-Learn and TensorFlow. Information about trainable (T) or non-trainable (NT) layers are provided for transfer learning.

Model	Parameters
Random Forest	criterion='log_loss', max_depth=7, max_features='log2', min_samples_leaf=1, min_samples_split=2, n_estimators=10, random_state=42
SVM	C=5, degree=1, gamma=0.01, kernel='poly', random_state=42
XGBoosting	booster='gbtree', eta=0.01, gamma=0, max_depth=3, min_child_weight=1, random_state=42
Logistic Regression	C=100, penalty='l2', solver='newton-cholesky', random_state=42
K-Nearest Neighbors	leaf_size=1, n_neighbors=1, p=1
Gaussian Naïve Bayes	var_smoothing=3.5111917342151277e-08
Neural Network	
NN (Transfer Learning)