Using Chat GPT for Malicious Web Links Detection

Thomas Kaisser[a] and Claudia-Ioana Coste[b]

Department of Computer Science, Babeș-Bolyai University, Mihail Kogălniceanu street, no. 1, Cluj-Napoca, Romania

Keywords: Malicious Web Links Detection, Machine Learning, Ensembles, OpenAI API, Chat GPT.

Abstract: Over the last years, the Internet has monopolized most businesses and industries. These outstanding advancements lead to the dangerous development of specialized threats employed to outsmart everyday users, collect personal data and financial benefits. One of the most relevant attacks is malicious web links, which can be inserted into private messages, emails, social media posts and others to deceive consumers and trick them into clicking. Present approach will classify links based on multiple manually extracted features. Then, we perform a feature importance analysis. Moreover on a smaller dataset, we employ OpenAI's models to classify and then add a new feature representing the Chat GPT classification. Thus, we manage to improve the overall performance of multiple machine learning methods. The first experiment considers only a Random Forest classifier but in the second one, we added thirteen other intelligent algorithms and ensembles constructed from the best performing ones. The best obtained accuracy (95%) is reached by the RF model on the whole dataset.

1 INTRODUCTION

The last decade was marked by an undeniable evolution of the online environment. According to Statista (Statista Research Department, 2024), in 2023 there were 93.09% of all European households having an Internet connection. Moreover, 2022 was irrevocably marked by the launch of the Chat GPT 3 by OpenAI, which proved to be one of the most performant general online chatbot. After it, multiple companies developed other Large Language Models (LLMs) for general usage, providing human-like assistance for everyday tasks. LLMs are implemented using the self-attention mechanism, which was created to make the model attentive towards specific input data that may be relevant later. Transformers are an encoder - decoder type of architecture, where multiple multi-headed attention layers are used (Vaswani et al., 2017). These types of layers will compute an attention score associated with each token, how relevant it is considering the context.

Even if Chat GPT and other popular LLMs (e.g., Gemini, Llama 2, Copilot, etc.) provide online, easy-to-use and fast functionalities to users for a large variety of redundant tasks (e.g., code debugging, content creation, brainstorming, explaining new concepts, etc.) it also introduced an easy manner to create harmful content to be spread online. The malicious content can be created by evading the OpenAI's guards (OpenAI, 2024b). An analysis on how Chat GPT is able to generate phishing attacks is tackled in (Roy et al., 2023), where nine examples of credentials theft attacks were deployed by impersonating 50 famous website brands (e.g., Facebook, Amazon, PayPal, etc.). The malicious strategies include: regular phishing, reCAPTCHA, vicious links presented as QR codes, Browser-in-the-Browser attack, iFrame injection (clickjacking), exploiting DOM classifiers, polymorphic URLs (Uniform Resource Locators), text encoding exploits and browser fingerprinting. All the attacks were implemented using Chat GPT's prompts with little to no human influence.

Present paper proposes to investigate the usage of Chat GPT models in detecting malicious tasks. Our contributions are the following:

- Build a baseline Random Forest (RF) model with multiple hand-crafted features and analyze their importance;
- Link classification using the OpenAI's Chat GPT 4 and 3.5-turbo and four different prompts;
- Enrich the baseline model with an additional feature representing the Chat GPT's classification;
- Experiment with multiple machine learning (ML) algorithms and ensembles.

This research paper is split into five chapters. We continue with the "Introduction" 1 section, then with

[a] https://orcid.org/0009-0001-0025-6415
[b] https://orcid.org/0000-0001-8076-9423

Kaisser, T. and Coste, C.
Using Chat GPT for Malicious Web Links Detection.
DOI: 10.5220/0013069200003825
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 20th International Conference on Web Information Systems and Technologies (WEBIST 2024)*, pages 425-432
ISBN: 978-989-758-718-4; ISSN: 2184-3252
Proceedings Copyright © 2024 by SCITEPRESS – Science and Technology Publications, Lda.

a section 2 reviewing the state-of-the-art concerning malicious web links detection. Next, we describe our empirical methodology with the steps taken towards achieving our objectives 3. The following section will describe our experiments, configurations and results 4, including comparisons with other approaches. Finally, we conclude this study and draw some future directions for research.

2 MALICIOUS WEB LINKS DETECTION

Malicious web links detection is often a binary classification or a multi-classification problem. To properly detect the link's class, scientists take into consideration a large variety of characteristics, extracted from the URL (i.e., content-independent) or from the web content (i.e., content-dependent). Most of the published approaches are working with ML, however there are ideas considering blacklists, anti-viruses, and complex network theory.

2.1 Content Independent Approaches

Content-independent solutions are employing just features extracted from the URL, such as lexical, computer network information based on Domain Name System (DNS), WHOIS or other external services. As example in (Oshingbesan et al., 2021) there are a total of 380 lexical features extracted from the URL (e.g., word2vec, N-grams, etc.). The approach compared ten ML algorithms such as Logistic Regression (LR), Linear Support Vector Machine (SVM), Decision Tree (DT), RF, Categorical Boosting, K-Nearest Neighbor (KNN), Feed Forward Neural Network (FFNN), Naive Bayes (NB), K-Means and Gaussian Mixture Model. The experiments are done on multiple datasets, which were aggregated from multiple sources. KNN was the best performing algorithm and word2vec features were not found to be relevant in the link classification.

With more types of characteristics extracted from the URL (i.e., lexical, DNS related and third-party information), (Mahdavifar et al., 2021) propose a KNN achieving 98.9% accuracy. The final model depicted the best thirteen features according to information gain. Additionally, it was proved that third-party data (e.g., domain age, geolocation, domain name, Alexa's rank, etc.) was more relevant in link classification. KNN was compared with SVM, Multi-layer Perceptron (MLP), Gaussian Naive Bayes (GNB) and LR. The experiments were conducted on a novel dataset with 400,000 benign samples, and 13,011 malicious.

One of a more traditional and old method of detecting the maliciousness of a link is using blacklists, which are public-available lists containing the malicious domains. These lists are periodically updated. Such a method was implemented in (Ma et al., 2009) as the first step of the labeling. Blacklist approaches have disadvantages, such as that they are not suitable for zero-days attacks. Moreover, web domains and web content are very dynamic in time. A malicious website could be taken down and a new safe website could be replacing it and vise versa.

2.2 Content Dependent Approaches

Content-dependent strategies operate with other features which can be extracted from the content of the web page, from Hypertext Markup Language (HTML), JavaScript (JS), Cascading Style Sheets (CSS) or media files, such as images, audios, videos, font files, etc.

In (Wejinya and Bhatia, 2021), there are added handcrafted content-based features besides lexical and host-based ones. Out of a total of 30 characteristics, the NB model works best with just 15 of them considered to be the most relevant. Similarly, in (Kumi et al., 2021) the model is enriched with content-based features extracted from the HTML or JS data. The classification is done using a data-mining algorithm, association classifier, which is achieving an accuracy of 95.8%. It is compared with other approaches and with other ML methods (e.g., LR, SVM, NB). The most important features proved to be the entropy of the domain name, JS tags, DOM functions and other information extracted from the JS files. Likewise, (Nagy et al., 2023) is proposing a similar idea with most attributes elected from the JS and HTML content. All features were passed through the chi-square test and depicted just the best 15 of them to form the detection model. The compared models include RF, NB, Convolutional Neural Network (CNN) and Long-Short Term Memory (LSTM), the best results being reached with NB (96.01% accuracy).

A more complex approach is done by (Rozi et al., 2021), where JS code was parsed and transformed into a graph, which was then passed through graph2vec and given as input to the classification model. The intelligent method was chosen based on multiple comparisons between: a neural network (NN), MLP, NB, LR, DT, Gradient Boosted Tree (XGB), and SVM with Radial basis function kernel.

Recently, there have been developed a novel solution with the rise of generative artificial intelligence as the one proposed in (Koide et al., 2023). LLMs are employed to help with the phishing/non-phishing link classification. The models used take into consideration multiple features: URL, HTML and text information collected from the screenshot of the website. The detection models were developed as text models and as multimodal ones, the last ones are able to process images as input. The proposed solution, ChatPhishDetector, is also checking the website for brand impersonation. The dataset on which the experiments were driven, contains 1,000 samples for each class. The LLMs were configured with two prompts, which asked for an explaination on why a certain class was chosen. The authors compared Chat GPT 4, 4vision and 3.5-turbo; Gemini Pro and Pro Vision; Llama-2.

Even before, cybersecurity companies were enriching their products with Generative AI. An example is VirusTotal's Code Insight (Quintero, 2023) utilizing Sec-PaLM, a proficient LLM provided by Google. The product helps cybersecurity specialists to faster analyze code, by translating the code into natural language. Another example is described in (Tushkanov, 2023) where a series of simple experiments are delivered with the task to classify phishing websites. The performance achieved is around 87% accuracy with a high false positive rate, which means that many websites were considered unsafe even though they were safe.

Still, there are not a large variety of solutions in using the capacities of LLMs, in different ways. Thus, we propose to enrich ML models with an supplimentary feature representing the classification done by a LLM.

3 EMPIRICAL METHODOLOGY

Herein, we present the methodology followed when experimenting with malicious web links detection task and Chat GPT. Present empirical study passes through the next stages:

- Dataset selection and preprocessing;
- Feature modeling;
- Chat GPT classification;
- ML models development;
- Comparisons.

3.1 Dataset Selection and Preprocessing

The depicted dataset can be accessed at (Siddhartha, 2021). It includes a total of 651,191 URLs, classified into four categories: 428,103 benign, 96,457 defacement, 94,111 phishing, and 32,520 malware. It successfully captures a large variety of web threats, aggregating data from multiple sources, such as ISCX-URL-2016 (Mamun et al., 2016), Faizan's github (Joerg, 2017), Phishtank (PhishTank, 2023), PhishStorm (Marchal et al., 2014) and "malwaredomainlist.com" (malwaredomainlist, 2010).

Even though we started with multi-classification we continued with binary classification since it was easier to compare with other approaches. All labels from the dataset including: phishing, defacement and malware were remapped to "unsafe". The benign samples were renamed to "safe". The dataset is not balanced and we consider this a proper and realistic representation on how many benign and malicious links there are in a real scenario.

3.2 Feature Modeling

The characteristics elected were based on the previous work done in the domain of malicious / benign link classification using manually-engineered features. Present experiment could be split into three stages:

- the simple stage - adding each feature one by one and observe if accuracy increases;
- the feature importance score - where characteristics were dropped based on their relevance score;
- covariance matrix - where the highly coupled features were eliminated.

The simple method, where starting from an initial set of features, we added one at a time to observe some improvements in accuracy score. We started with an initial set of features S^i. The baseline model on which this experiment was performed is RF. After running the experiments, we developed the final set of features S^f. All attributes were extracted based on the URL and they fall into the lexical and host-based categories. In table 1 there can be found a compilation of all features tried and added into the model. The gray rows indicate the starting set of characteristics for the model together with a brief explanation. Pink and white rows contain the added features that we have selected to enrich the model and improve its performance.

The next experiment considers the importance scores, which are calculated as the mean decrease in impurity across all trees within the RF model. The computation is automatically done by the Sklearn library (Pedregosa et al., 2011). In this stage, we further eliminate the features with a small contribution to the classification.

In the final stage, the covariance matrix was computed and based on it, the most redundant features were dropped. If there are multiple features highly correlated, the one having the lowest importance score will be eliminated.

Table 1: The features selected and their descriptions.

Feature name	Description
has IP address	checking if the URL contains an IP address
no. full stops	counting the ”.” sign
no. ”@” sign	counting the ”@” sign
Google Index	checks if the URL is indexed by Google service, offering an indication on the legitimacy of the URL (Vikramaditya, 2024)
no. embedded domains	count of the domains found within the URL (separated by ”//”)
no. directories	counting the directories found in the URL path (separated by ”/”)
length of the URL	total length of the URL
no. digits	counting the digits found in the URL
no. special chars.	count the special characters (”/”, ”%”, ”#”, ”&”, ”=”, ”?”) found in the URL
Shannon entropy	computed on the network location of the URL (including the domain, port or subdomains if there is any) (Lin, 1991)
longest token FQDN	the length of the longest token found in the Fully Qualified Domain Name (FQDN) considering the network location
no. vowels	counting the vowels (”a”, ”e”, ”i”, ”o”, ”u”) found in the URL path
query length	length of the query string (the URL part between the ”?” and the fragment sign ”#”)
is HTTPS	checking the presence of the HTTPS protocol in the URL
no. phishing words	counting general phishing words (”webscr”, ”secure”, ”banking”, ”ebayisapi”, ”account”, ”confirm”, ”login”, ”signing”) (Alabdulmohsin et al., 2016)
port indicator	checking the presence of the port number within the URL

3.3 Chat GPT Classification

We employ two OpenAI models: GPT 3.5-turbo and GPT 4, using OpenAI Python API (OpenAI, 2024a). Four simplistic prompts were engineered and they are exemplified in Table 2. The first one is the most simple one. The second one and the third one are inspired by the Zero-Shot technique detailed in (Kojima et al., 2022). Lastly, the forth prompt considers the Chain of Though technique described in (Wei et al., 2022).

Due to financial limitations and the costs charged by the OpenAI's models we proposed to run our tests on just 1000 random links. Depending on the prompt type, GPT-3.5-turbo has a cost between 0.12$ and 0.27$ per testing set, while GPT-4 is more expensive with 1.72$ - 9.06$. The longer the prompt the more expensive it was to test the model. There will be eight combinations of prompts and models which will be tested and compared. Then, we enriched the RF model and the other ML models with an additional feature representing the LLM's labeling. By doing so, we should achieve a greater performance. The models using Chat GPT features will be trained and tested on a total of 1000 links randomly sampled from the dataset (Siddhartha, 2021). 800 of the web links will be used for training and 200 of them for testing.

3.4 ML Models Development

The baseline model used is a RF implemented in Sklearn library (Pedregosa et al., 2011) and a manual parameter calibration after multiple runs. The rest of the used models are the following: RF, XGB, ADA, DT, LR, Gaussian NB (GNB), Multinomial NB (MNB), Complement NB (CNB), Quadratic Discriminant Analysis (QDA), Linear Discriminant Analysis (LDA), Passive Aggressive Classifier (PAC), SVM and KNN. All algorithms were utilized from SKlearn Python library (Pedregosa et al., 2011). The parameters for all models were the default ones.

Moreover, we employ an ensemble using the three of the best performing algorithms. The ensemble is developed with a Voting Classifier from (Pedregosa et al., 2011). Multiple types of ensembles are tried with different voting mechanisms (”hard”, ”soft”) and different weights generated based on the accuracy score of the individual classifiers. For all tests the dataset was randomly split into 80% training and 20% testing.

3.5 Comparisons

Our proposed approach is combining ML models and ensembles with a Chat GPT feature. This idea will be compared with other similar literature solutions using the same dataset (Siddhartha, 2021). All records from the dataset are used in (Zhang and Yan, 2023), in (Shetty et al., 2023) and in (Coste, 2024). Additionally, there are other literature papers using the same dataset but on a subset of 40,000 links. Thus, to preserve a fair comparison, regarding the number of records, we consider just these three solutions relevant for comparisons.

Table 2: The prompts tried for both Chat GPT models.

No.	Prompt
1	Check if the url is safe or not, respond with SAFE for safe url and UNSAFE for unsafe url
2	Check if the url is safe or not, respond with SAFE for safe url and UNSAFE for unsafe url. Try to look at the entropy of the domain, the length of the domain, the number of the special characters, the longest token in the url.
3	Check if the url is safe or not, respond with SAFE for safe url and UNSAFE for unsafe url. Try to look at the entropy of the domain, the length of the domain, the number of the special characters, the longest token in the url. Examples: 1. http://www.ikenmijnkunst.nl/index.php/exposities/exposities-2006,unsafe 2. http://peluqueriadeautor.com/index.php?option=com_virtuemart&page=shop.browse& category_id=31&Itemid=70,unsafe 3. movies.yahoo.com/shop?d=hv&cf=info&id=1800340831,safe 4. duckduckgo.com/1/c/Roman_Catholic_cathedrals_in_Canada,safe 5. alexpay2.beget.tech,unsafe 6. http://worldoftanks.ru/ru/content/guide/payments_instruction/mobile-payments-rostelekom-ural-utel/,safe 7. http://www.artedesignsas.it/catalogo.html?page=shop.browse&category_id=14,unsafe
4	Check if the url is safe or not, respond with SAFE for safe url and UNSAFE for unsafe url. Look at these features: - Number of directory levels - Length of the URL: 38 characters - Number of special characters (from ”/%#&=?”) - Shannon entropy of the domain - Length of the longest token in the FQDN - Number of dots in the URL - Number of vowels in the path Examples: friars.com/sports/m-baskbl/archive/prov-m-baskbl-2003.html is Safe because the values of the extracted features are: [4, 58, 4, 0.0, 0, 2, 10] http://www.martin-busker.de/administrator/help/en-GB/css/Facture/ c4d12146ebce8e1684d3542308399779/8fa39 dab95edb1b676b638a672278eae/particuliers-45636.php is Unsafe because the values of the extracted features are: [8, 153, 10, 3.78, 13, 3, 25]

4 EXPERIMENTS AND RESULTS

In the following section, we describe the feature importance experiment and how by adding the GPT's prediction into different ML models they are able to improve prediction in malicious web links detection.

4.1 Feature Importance Results

All feature importance experiments could be split into 3 stages (i.e, simple stage, feature importance stage and covariance matrix stage) as previously described in the Methodology section 3.2. The first assessment started with a simple RF model and a predefined set of characteristics selected from other solutions from the state-of-the-art. The initial features are marked with pink in Table 1 from Methodology section 3.2. To the initial set, there were added one at a time the rest of the features to observe an improvement in accuracy. Features marked with white (see Table 1) did not improve the performance of the RF, and the experiment needed more time to extract these features. The gray ones added a significant increase in metrics.

The second experiment took into consideration the feature importance score computed by the Sklearn RF model (Pedregosa et al., 2011). Current stage includes five runnings and the features obtaining the lowest score were dropped in an iterative method such that the evolution of the accuracy could be observed as well. Finally, the following features were dropped due to their low score: IP address, number of ”@” sign, Google index, the number of embedded domains and of directories. With the final set of features (i.e., number of full stops, number of directories, length of the URL, count of special characters, Shannon entropy, length of the longest token in FQDN and number of vowels), the RF model achieved 93.78% accuracy.

Further, the final stage will select features considering the covariance matrix. There was observed a high correlation between the length of the URL, the count of digits and the count of special characters. This correlation is normal to happen since both the number of digits and special characters are included in the total length of the web link. If the URL is longer it is a high probability that the number of special characters and digits is larger. Therefore, considering the importance score, we dropped the number of digits to avoid this redundancy.

4.2 Using Chat GPT for Link Classification

OpenAI's models were used for a standalone classification using two models (GPT-3.5-turbo and GPT-4) and four prompts as described in Table 2. Then, based on the best combination of a model and a prompt, a new feature was added into the baseline RF model. The tests were done on 1000 randomly sampled links.

Considering the obtained performance, GPT-4 outperformed GPT-3.5-turbo as it can be observed in Table 3. The best model was GPT-4 with the third prompt, reaching 65% accuracy. The best model with GPT-3.5-turbo was achieved with the second prompt. Regarding the cost, GPT-3.5-turbo is considerably cheaper than GPT-4. Moreover, if we compare the best classification with GPT-3.5-turbo and the best of GPT-4, we can observe a small difference in performance but a significant one regarding cost.

Afterwards, we added a new feature ("openAICallCheck") in the previously developed RF model with the final aim to further increase its accuracy. This feature represents the link classification as done by the GPT-4 model with prompt 3, our best GPT configuration. We observed that on the evaluating set of 1000 links, the RF accuracy rose from 88.8 to 89%. The metrics were computed for five different dataset splits. Thus, even though GPT models do not have a high accuracy on their own, using their classification as input to a ML model, there may be a modest increase in performance.

Table 3: The performance of the chat GPT models (3.5-turbo and 4) for link classification (testing set).

Prompt	Acc.(%)	Precision	Recall	F1	Cost
GPT-3.5-turbo					
1	48.5	65	54	40	0.12 $
2	63	66	65	62	0.15 $
3	54.5	65	58	51	0.24 $
4	44.5	64	51	34	0.23 $
GPT-4					
1	60	64	60	60	1.72 $
2	61	67	64	60	3.15 $
3	65	65	65	65	7.35 $
4	64	66	65	64	9.06 $

4.3 Performance of the Other ML Models

Taking into consideration the potential of the "openAICallCheck" feature, we propose to further experiment with multiple ML algorithms and ensemble models. Table 4 details the accuracy scores for all ML models employed and they were computed on the testing set on five dataset splits. The first two columns contain the accuracies without the openAI's feature and with it. The final column has the accuracies achieved on the whole dataset, which will be used for reference and to observe if the ML models generalize well. These experiments do not take into account OpenAI's prediction due to financial limitations. The total cost for all the 651,191 links would have reached 4,700 $ by using GPT-4 and prompt 3. It can be observed that most algorithms have a modest improvement in accuracy when adding the "openAICallCheck" feature. Usually, there was a slight decrease in accuracy was noted for XGB, DT, QDA, and LDA. However, for the PAC algorithm the GPT's information proved to be rather detrimental. This may happen because PAC is an online learning algorithm, where the training set is processed sequentially and the model is updated in the same manner. PAC is suitable for large data while small amounts of data may not be enough. For the rest of the ML methods (i.e., RF, ADA, LR, NB algorithms, MLP, KNN, and SVC) there can be seen an increase between 0.2 and 1.1, which we consider to be relevant.

Table 4: The accuracy of the all ML models (testing set).

Model	Acc.	Acc. (with GPT)	Acc. (all)
RF	88.8	89	95
XGB	93.4	93.3	92.86
ADA	92.6	93.3	91.08
DT	91	90.9	94.35
LR	87.9	88.3	85.43
GNB	85.1	86.2	84.56
MNB	86.9	87.8	84.55
CNB	86.6	86.9	84.47
MLP	90.8	91.2	93.3
KNN	87.8	88.5	93.38
QDA	88.2	87.3	84.67
LDA	88.1	87.9	84.99
PAC	87.7	72.1	83.84
SVC	86.9	87.9	87.33

Regarding generalization, RF, DT, MLP and KNN prove to achieve a greater performance when trained on more data. Even though the rest of the algorithms generalize well, there is not a significant drop in accuracy when trained on all 651,191 records. It is definitely a case of overfitting and it should be investigated more.

Overall, the best performing ML methods are XGB, ADA and DT, which will be depicted to form the heterogeneous ensembles. The weights represent the accuracy score obtained by the models in an individual setting. Ensembles were calibrated considering the voting mechanisms (i.e., soft or hard) and by adding weights or not. The experiment was conducted in the same configuration as the one for the single models. While for most ensembles, adding the GPT's feature was detrimental, for the no-weights soft-voting ensemble it was observed a light increase in accuracy (0.1%). This may be due to the fact that hybrid models need more data to be effective. This is sustained by our results on the whole dataset, where all ensembles significantly improved in accuracy (1-

5%). Thus, testing the "openAICallCheck" feature on just 1000 links may not be enough to properly train an ensemble model. Moreover, by comparing the ensemble with the individual results of the classifiers on 1000 URLs, it did not lead to an increase in accuracy. Although, on the whole dataset, the ensemble achieves rather better results (e.g., 94.29%).

4.4 Comparisons and Discussion

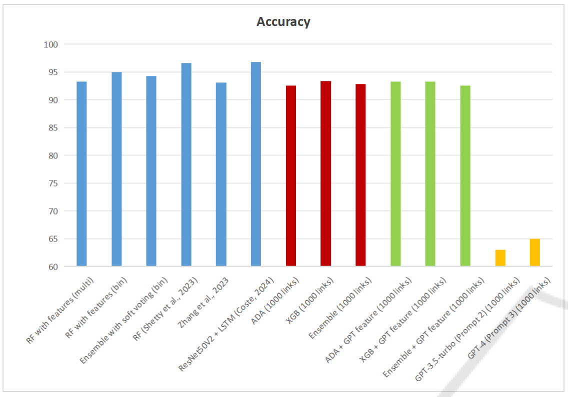

Figure 1: Comparisons with other literature approaches.

Our purpose was to investigate the support of using Chat GPT in classifying web links as malicious or benign. This idea was tested with multiple intelligent algorithms including ensembles. The best performing algorithm on the whole dataset (Siddhartha, 2021) was RF with 95% accuracy. On the smaller dataset formed by 1000 randomly sampled links, the best accuracy was obtained by XGB, closely followed by ADA. Also, both of these models were the most accurate when adding the GPT's prediction as a feature. The best ensemble was formed by XGB, ADA and DT and it is characterized by a hard voting mechanism and weights. This ensemble has the highest accuracy rate on the 1000 links dataset. On the whole dataset, the soft-voting ensemble has a higher accuracy rate. Figure 1 presents all our best models. The blue bars represent the classification for the whole dataset even if it is a binary classification or it is a multi-classification. The red ones mark the results obtained on the smaller dataset of 1000 links. The green bars signify the accuracy scores reached by our best algorithms on the smaller dataset, but adding the GPT's classification to the model. The orange bars present the best accuracies obtained from the standalone OpenAI's models.

Considering the whole dataset, there can be observed that our models outperform the solution provided by (Zhang and Yan, 2023), but unfortunately, they are behind with other more performant models from (Shetty et al., 2023) and (Coste, 2024). Regarding the smaller dataset using just hand-crafted features, the ensemble model does not reach a better accuracy rate compared to the individual model, which we would have expected. By comparing with the models using the "openAICallCheck" characteristic, we can observe a small improvement for ADA, but for the ensemble or XGB, the accuracy has a tiny decrease. Nevertheless, set side by side with the standalone GPT's models (i.e., gpt-3.5-turbo and gpt-4), our models certainly are a better solution. We chose to not include the solution from (Koide et al., 2023) in the Figure 1 because the comparison would not be fair since the dataset is different. Still, their approach is much more accurate with higher metrics obtained and multiple LLMs engines and tasks. All in all, our approach can pave the way for novel solutions by extracting features from OpenAI's models to advance the classification of malicious links.

5 CONCLUSIONS AND FUTURE WORK

Malicious web links account for multiple security attacks directed against inexperienced users and can lead to drive-by-downloads, credentials theft, impersonating brands and deceiving people. Present paper proposes to tackle the application of OpenAI's models (i.e., GPT-3.5-turbo and GPT-4) to counteract web-malware. Our experiments contain a feature importance analysis on web links with multiple hand crafted features. Then, a large variety of intelligent methods including ML models and ensembles were extended with a feature considering the GPT's prediction. The experiments proved that by appending OpenAI's prediction of a link as a new feature model it can slightly improve the accuracy of most algorithms. The best models achieve 94-95% accuracy on the whole dataset.

Using the capabilities provided by LLMs could lead to major improvements regarding cybersecurity. For future work, we propose to add other Chat GPT related features into the ML classification models such as an explanation on why a link is malicious, its domain similarities with other brands on the Internet, etc. As well, other engines, such as Gemini, Llama, GPT-4o etc. should be use for comparisons. Additionally, by using transfer learning, a large language model could be trained specifically on the task of malicious web links detection. Moreover, the problem should be addressed on a larger scale since links need to be checked in a continuous and fast way not to interfere with the online environment.

REFERENCES

Alabdulmohsin, I., Han, Y., Shen, Y., and Zhang, X. (2016). Content-agnostic malware detection in heterogeneous malicious distribution graph. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 2395–2400.

Coste, C. I. (2024). Malicious web links detection based on image processing and deep learning models (accepted for publication). In *The 23rd IEEE/WIC International Conference on Web Intelligence and Intelligent Agent Technology*.

Joerg, S. (2017). Using-machine-learning-to-detect-malicious-urls. faizan dataset link (Retrieved: August 22, 2024).

Koide, T., Fukushi, N., Nakano, H., and Chiba, D. (2023). Detecting phishing sites using chatgpt. *arXiv preprint arXiv:2306.05816*.

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. (2022). Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Kumi, S., Lim, C., and Lee, S.-G. (2021). Malicious url detection based on associative classification. *Entropy*, 23(2):182.

Lin, J. (1991). Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory*, 37(1):145–151.

Ma, J., Saul, L. K., Savage, S., and Voelker, G. M. (2009). Beyond blacklists: learning to detect malicious web sites from suspicious urls. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1245–1254.

Mahdavifar, S., Maleki, N., Lashkari, A. H., Broda, M., and Razavi, A. H. (2021). Classifying malicious domains using dns traffic analysis. In *2021 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, pages 60–67. IEEE.

malwaredomainlist (2010). Malware domain list. URL to malwaredomainlist (Retrieved: August 22, 2024).

Mamun, M. S. I., Rathore, M. A., Lashkari, A. H., Stakhanova, N., and Ghorbani, A. A. (2016). Detecting malicious urls using lexical analysis. In *Network and System Security: 10th International Conference, NSS 2016, Taipei, Taiwan, September 28-30, 2016, Proceedings 10*, pages 467–482. Springer.

Marchal, S., François, J., State, R., and Engel, T. (2014). Phishstorm: Detecting phishing with streaming analytics. *IEEE Transactions on Network and Service Management*, 11(4):458–471.

Nagy, N., Aljabri, M., Shaahid, A., Ahmed, A. A., Alnasser, F., Almakramy, L., Alhadab, M., and Alfaddagh, S. (2023). Phishing urls detection using sequential and parallel ml techniques: Comparative analysis. *Sensors*, 23(7):3467.

OpenAI (2024a). Openai platform. URL to malwaredomainlist (Retrieved: August 22, 2024).

OpenAI (2024b). Usage policies. Link to article (Retrieved: August 22, 2024).

Oshingbesan, A., Okobi, C., Ekoh, C., Richard, K., and Munezero, A. (2021). Detection of malicious websites using machine learning techniques. *preprint*, none(none):1–5.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

PhishTank (2023). PhishTank - Out of the Net, into the Tank - Developer Information. PhishTank website (Retrieved: August 22, 2024).

Quintero, B. (2023). Introducing virustotal code insight: Empowering threat analysis with generative ai. Link to article (Retrieved: August 22, 2024).

Roy, S. S., Naragam, K. V., and Nilizadeh, S. (2023). Generating phishing attacks using chatgpt. *arXiv preprint arXiv:2305.05133*.

Rozi, M., Ban, T., Kim, S., Ozawa, S., Takahashi, T., and Inoue, D. (2021). Detecting malicious websites based on javascript content analysis. In *Computer Security Symposium 2021*, Dubrovnik, Croatia. Computer Security Symposium 2021.

Shetty, U., Patil, A., and Mohana, M. (2023). Malicious url detection and classification analysis using machine learning models. In *2023 International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT)*, pages 470–476. IEEE.

Siddhartha, M. (2021). Malicious urls dataset. Kaggle - Malicious URLs dataset (Retrieved: August 22, 2024).

Statista Research Department (2024). Household internet access in the european union 2023. Link to article (Retrieved: August 22, 2024).

Tushkanov, V. (2023). Investigating chatgpt phishing detection capabilities. Link to article (Retrieved: August 22, 2024).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Vikramaditya, N. (2024). Googlesearch-python. Link to library (Retrieved: August 22, 2024).

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. (2022). Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Wejinya, G. and Bhatia, S. (2021). Machine learning for malicious url detection. In *ICT Systems and Sustainability*, pages 463–472. Springer, Singapore.

Zhang, L. and Yan, Q. (2023). Detect malicious websites by building a neural network to capture global and local features of websites. *Research Square*.