

# A methodology combining NLP, Machine Learning, and LLM for the Detection of Phishing Emails

Danilo Rosati<sup>1,†</sup>, Alessandro Aldini<sup>1,\*,†</sup>

<sup>1</sup>University of Urbino Carlo Bo, Urbino, Italy

## Abstract

This work explores the combined use of various AI-based methods for detecting and classifying phishing emails. Specifically, we consider Random Forest, Support Vector Machine, and XGBoost machine learning (ML) algorithms, alongside natural language processing (NLP) techniques, and the large language model (LLM) Gemini. We combine all these methods in a pipeline and, through an empirical analysis based on a publicly available, balanced dataset, we compare the various techniques and emphasize the potential advantages and disagreements arising from the integration of ML and LLM.

## Keywords

Phishing, Large Language Models, Machine Learning Classifiers, Natural Language Processing

## 1. Introduction

In recent years, phishing attacks have emerged as one of the most pervasive and malicious threats in the social engineering landscape. This ever-evolving form of cybercrime targets and persuades users to divulge sensitive information such as banking details and login credentials [1] by exploiting human vulnerabilities that can potentially be found in every user, such as lack of knowledge, stress, and scarcity of time [2]. Email has become the most common means of conducting phishing attacks [3, 4]. Despite continuous updates in methodologies to counter these threats, the effectiveness of current solutions is often inadequate, as evidenced by the exponential increase in email phishing attacks in recent years [1]. This persistent threat underscores the need to develop and implement effective detection systems to be able to protect users and organizations from potential financial damage and theft of sensitive data.

Numerous techniques for filtering phishing emails have been proposed over the years. These range from systems based on denylists of domains and IP addresses [5, 6, 7], to advanced authentication solutions, client-side tools, and user awareness initiatives [8]. However, the phishing problem still remains definitively unsolved, making the search for more comprehensive and effective solutions imperative. Among the most recent approaches to address this challenge are the use of AI-based techniques, ranging from Natural Language Processing (NLP) to Machine Learning (ML) and Large Language Models (LLM). NLP offers the ability to analyze the textual content of emails, identifying linguistic patterns typical of fraudulent communications [9, 3]. ML enables learning from the characteristics of legitimate and phishing emails, developing patterns that can automatically classify new instances [10]. The integration of these two disciplines has shown significant potential in improving the accuracy and effectiveness of phishing detection systems [1]. Moreover, more recent developments propose the use of LLMs to classify emails and detect potential phishing attacks [2].

In this very diverse context, the present work focuses on the combined use of NLP techniques, ML algorithms, and LLM for the detection of phishing emails, in order to compare their performance and the potentialities of their integration from the perspective of the user. On the one hand, we consider Random Forest (RF), Support Vector Machine (SVM), and XGBoost algorithms as ML techniques; on the other hand, in a subsequent phase, integration with an LLM is proposed to empirically compare

*Joint National Conference on Cybersecurity (ITASEC & SERICS 2026), February 09-13, 2026, Cagliari, IT*

\*Corresponding author.

†These authors contributed equally.

✉ d.rosati2@campus.uniurb.it (D. Rosati); alessandro.aldini@uniurb.it (A. Aldini)

ORCID 0000-0002-7250-5011 (A. Aldini)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

the results with respect to explainability-related aspects. Hence, the main contribution of this paper is a framework based on a pipeline of different techniques, including NLP, ML, and LLM, which is proposed to (1) support the user in detecting and understanding the nature of phishing emails, and (2) investigate not only the efficiency of different methods, but also the potential differences arising from their combined use. As a result, we present performance metrics derived from the analysis of a real-world dataset over the implementation of the proposed pipeline. As an additional contribution, a feedback mechanism supporting continuous learning is added to the pipeline.

The paper is organized as follows. Section 2 discusses the methodology we followed to combine several different techniques in our framework, while Section 3 presents the related implementation. In Sections 4 and 5, we present the results of our evaluation and a comparison with related work, respectively. Section 6 concludes the paper with a discussion about the future developments of this framework.

## 2. Background and proposed methodology

In this section, we describe the methodological approach followed for the detection of phishing emails: from extracting the textual content of the emails and, if present, the included links, to capturing the linguistic and semantic features using NLP techniques, and then to the final classification using ML and LLM.

### 2.1. Extraction and preprocessing

The email content extraction and preprocessing steps are critical to the preparation of the data that will be used in the subsequent analysis and classification steps for phishing detection. In fact, the effectiveness of machine learning systems depends on the quality and preparation of the datasets [11].

The first step is to extract the relevant content of the email, which mainly includes the body of the text, headers, and any links present [2]. The following preprocessing stage is then aimed at reducing noise (including, e.g., unnecessary spaces and blank lines) [12] and preparing the data for analysis (by replacing or removing specific html elements present in the .eml file, such as < link > tags) [2].

A fundamental component of our phishing detection methodology is concerned with the analysis of links, which are often present in emails with the purpose of directing victims to counterfeit websites. Link analysis combines several aspects and approaches:

- Use of denylists: this is a well-established approach aimed at checking the presence of URLs previously identified as malicious; however, the effectiveness of this approach depends on the timeliness with which the denylists are updated [5, 2].
- Presence of URLs directly expressed as IP addresses: this is a clue of a technique often used to mask the true destination of the link [13].
- Domain age: the analysis on the age of the domain to which the link points can detect if it is a newly created website, a common practice for phishing sites [2].

### 2.2. Natural Language Processing

The integration of NLP techniques aims to improve the quality of the extracted data and obtain additional features (characteristics) that can improve the classification process. NLP is a field of artificial intelligence whose main goal is to enable computers to interpret and generate human language in a meaningful way. This includes the ability to analyze the meaning of words, sentences, and texts, including extracting useful information and responding appropriately [3, 1]. The main NLP phases are:

- Tokenization: splitting of the text into individual words or units (tokens) [7].
- Stop word removal: common, often uninformative words are removed from text data for efficiency and accuracy improvement [14].

- Stemming and lemmatization: reduction of the words to their base or root forms for uniformity purposes (stemming uses basic rules to remove suffixes, while lemmatization uses context-aware rules based on dictionary and morphological analysis) [7].
- Removal of punctuation and special characters [15].
- Lexical feature extraction: identification of relevant word-related characteristics, especially those that could be typical in phishing emails [13].
- Word embeddings: representation of the words as numerical vectors, capturing semantic relationships between words [12, 16].

At the end of the NLP operations, the output of the word embedding phase is used to feed the classification algorithms.

### 2.3. Classification models

The core of the classification is given by a machine learning classifier, for which we propose three different models based on the following approaches: Random Forest (RF), Support Vector Machine (SVM), and XGBoost.

RF denotes a class of algorithms based on decision trees. Basically, an RF model is made of two components:

- A collection (forest) of decision trees where each tree, trained on a random subset of the data and using a random subset of the available features, will provide a given prediction output [11].
- A voting mechanism or aggregation of results, which combines the results of each tree to make the final decision on a given instance. The final prediction of the RF is given by the majority vote [3].

RF models are particularly effective due to their ability to handle a large number of features and their robustness against individual errors [17].

SVM denotes a supervised ML algorithm for linear or non-linear classification, which aims at finding an optimal hyperplane (or set of hyperplanes) maximizing the margin between different classes in a high-dimensional space [10]. Nonlinear SVMs use the kernel functions [18], which allow the classification problem to be mapped from the original feature space to a higher feature space in which a linear separation is possible [19].

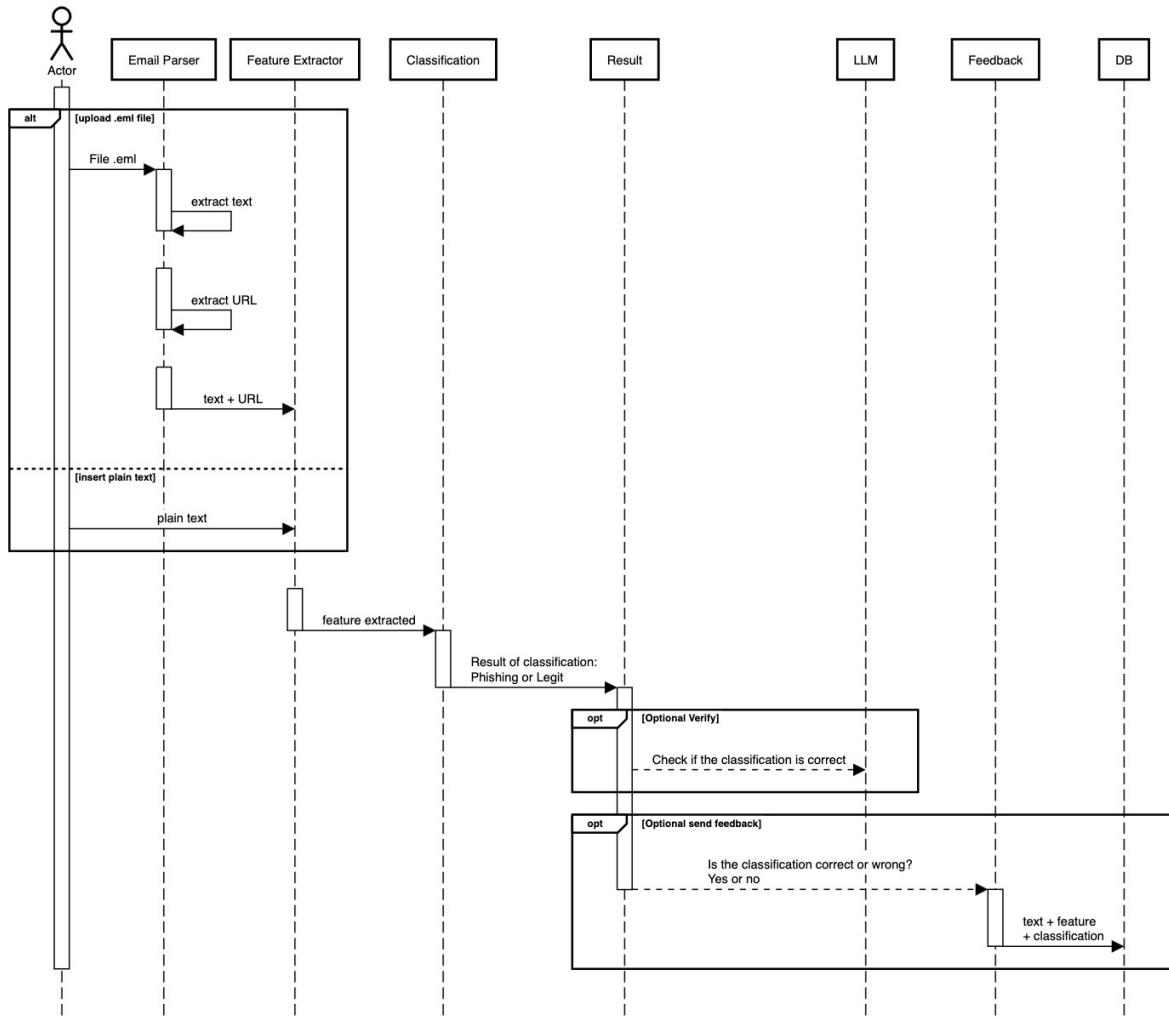
XGBoost is a refined and optimized version of the Gradient Boosting algorithm, designed to offer high performance in terms of accuracy, efficiency, and scalability [10]. The architecture of XGBoost is based on decision tree sequential learning, where each tree makes its own prediction and the final prediction results from the sum of the individual predictions. Each tree that is added to the model attempts to minimize a loss function, using a gradient descent method [20].

### 2.4. Integration with LLM

Innovative systems have been developed in recent years, such as APOLLO [2], which employs GPT-4 to classify emails as phishing or legitimate. LLMs offer a significant advance in the field of NLP as they are models that are trained on huge datasets, acquiring a vast amount of knowledge and the ability to identify complex patterns. Hence, they have proven very useful in classifying phishing emails by achieving very high levels of accuracy [2]. In the following sections, we will show how LLMs can be integrated into our classification process.

## 3. Implementation

By combining the techniques discussed above, in this section, we present the solutions adopted for data extraction, link analysis, and the integration of NLP tools, ML classifiers, and an LLM-based verifier.



**Figure 1:** System architecture and data flow

The system we propose [21] is structured into a pipeline of functional modules and has been developed in Java. In particular, we employed the libraries Weka [22] and XGBoost4J [23] (for the implementation of the classifiers), the API Google Cloud Natural Language [24] (for the analysis of the email content and writing style), and the model gemini-2.0-flash-lite [25] (for the integration with the LLM tool). Moreover, we used the framework Spring Boot [26] for the web interface.

The project is structured into functional modules as illustrated in Figure 1.

### 3.1. Email parser

The email parser module processes incoming emails with the aim of extracting relevant information from them. Extraction is done through the use of the Jakarta Mail library [27], which provides a comprehensive set of features for handling emails in standard format. The library allows the extraction of both plain text and HTML content text. The extracted text is then normalized through the use of the Jsoup library [28], which removes irrelevant HTML tags, detects hyperlinks present in the body of the email, and eliminates blank lines. The final output of the parser is the normalized textual content of the message and a list of all identified URLs that are then passed to the next module.

### 3.2. Feature extraction

In this phase, semantic and structural features are extracted from the data by using several different NLP techniques. The first and fundamental step in the feature extraction process is provided by BERT (Bidirectional Encoder Representations from Transformers), which is a pre-trained tool developed by Google. The distinguishing feature of BERT is its bidirectionality: unlike models that analyze text in only one direction (from left to right or vice versa), BERT is able to consider the full context of a word by looking at both those that precede and follow it [16].

For the use of BERT within our system, we have implemented an API written in Python, which takes as input the text of the email and then returns a 768-dimensional embedding vector. This vector captures the semantic characteristics of the text, representing an email as a point in a high-dimensional vector space, where similar emails are close together. The vector obtained by BERT is enriched with additional features extracted by:

- Sentiment Analysis: the Google Cloud Natural Language API takes text as input by returning a polarity score (Sentiment Score), which varies within a range [0 very negative, 1 very positive] and a confidence score (Sentiment Magnitude) representing the intensity of emotion.
- Link Analysis: binary features are added in cases where links are present, they use IP addresses, and they include non-ASCII characters.
- Keywords detection: binary features are added if the text has or does not have typical phishing words.

Finally, all extracted features are combined into a single vector used later as input for the classifiers.

### 3.3. Classification

The system uses three different ML classifiers, which can be used individually or simultaneously to compare results. For Random Forest, we assume a number of trees equal to 100, with no limitation on the maximum depth, and random feature selection for each node. For SVM, we adopt the Sequential Minimal Optimization algorithm with the Radial Basis Function kernel. This choice allows for modeling nonlinear decision boundaries, making it adequate to work on high-dimensional textual embeddings. For XGBoost, the model is configured for binary classification, taking advantage of boosting capabilities to improve accuracy and reduce overfitting.

### 3.4. LLM integration

In order to enhance the analysis of suspicious emails and provide a second level of verification, the system integrates Gemini. The module, therefore, offers the possibility of leveraging an independent verification mechanism. In addition to the response based on ML classifiers, Gemini offers an external opinion by describing in a way that is understandable even for non-technical users, why a particular email might be classified as phishing. The aim of this integration is also to compare the results of the different classification techniques. Interaction with Gemini is done through prompt engineering, which is dynamically customized based on the data obtained during the analysis; an example can be seen in Figure 2.

### 3.5. Feedback

At the end of the classification process, the system offers the user the opportunity to provide explicit feedback on the result obtained. This mechanism is particularly useful because, as we will see, ML and LLM might disagree. Hence, it is worth contributing to the construction of an incremental dataset for future model retraining. In practice, each feedback received is saved as a JSON file in a database hosted by Google Cloud Storage and includes: the text of the email, the embedding vector generated (BERT + NLP features), the classifier used for prediction, and the label predicted by the models.

Imagine you are a cybersecurity expert tasked with classifying an email as phishing or legitimate. The email has already been classified as PHISHING by the Random Forest classifier, but you must not be influenced by this classification as the classifier has a margin of error.

[CONTENT OF THE EMAIL]

New Ticket Sent Your ticket has successfully been sent to the store. \*Date Opened\* Monday, 03 July 2023 16:27 \*Store\* RetroAccounts View Ticket ( [https://retroaccounts.sellpass.io/tickets/\\*\\*\\*\\*?token=\\*\\*\\*\\*\\*](https://retroaccounts.sellpass.io/tickets/****?token=*****) ) This email was sent from a notification-only address that cannot accept incoming emails. Please do not reply to this message.

[URLS FOUND IN THE EMAIL]

- [https://retroaccounts.sellpass.io/tickets/\\*\\*\\*\\*?token=\\*\\*\\*\\*\\*](https://retroaccounts.sellpass.io/tickets/****?token=*****)  
-<http://cdn.mcauto-images-production.sendgrid.net/458add0cb793bde1/dec61d4b-feel-4e02-9348-a9025bb9373b/442x88.png>

[TECHNICAL INFORMATION] The email has been analyzed using multilingual BERT embedding. The embedding is a vector of 774 elements. The first 768 elements are the BERT embedding values, while the last 6 elements are specific features:

1. Presence of URL: Yes (1.00)
2. Presence of URLs with IP addresses: No (0.00)
3. Presence of URLs with non-ASCII characters: No (0.00)
4. Presence of typical spam words: No (0.00)
5. Sentiment score (Natural Language API): 0.96306. Sentiment magnitude (Natural Language API): 0.1180

REQUEST: Provide a detailed analysis of the email (up to 300 words), considering all the provided elements. Your analysis should include:

1. Whether or not you agree with the initial classification and why.
2. Suspicious elements or phishing indicators present in the text.
3. Analysis of the URLs (if any): Are they legitimate or suspicious?
4. Considerations on the extracted technical features.
5. Final conclusion: Do you classify this email as PHISHING or LEGITIMATE?

Respond in a structured format with headings for each section of your analysis. Keep the analysis concise but complete

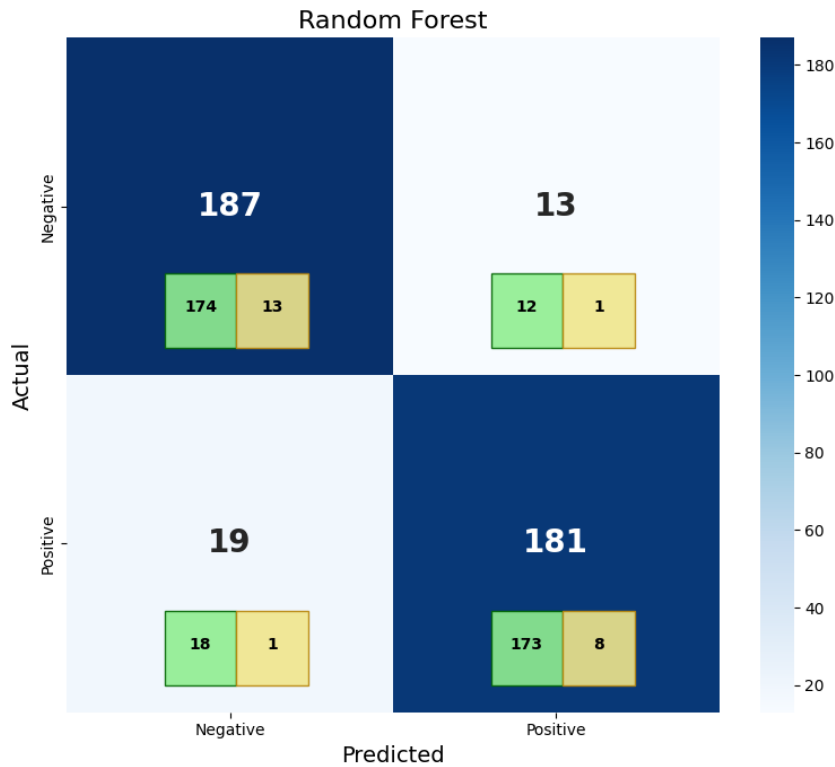
**Figure 2:** Gemini query based on the feedback reported by the ML classifier.

## 4. Evaluation

In this section, we present the results obtained by the ML models implemented to classify emails as either phishing or legitimate. For evaluation purposes, the three classification models have been trained on a balanced dataset (50% phishing, 50% legitimate) of 2000 emails [29]. The results for each model have been obtained on a validation set of 400 emails (equal to 20% of the training set) taken from the same source [29] and including new samples not included in the training set.

In order to evaluate the performance of the adopted classification models, as usual, we employ the corresponding confusion matrices, which are a fundamental tool for understanding the behavior of a classifier in detail. Each row of the matrix is representative of the actual labels (i.e., the correct class of the data), whilst each column indicates the class that the model has predicted. Then, denoted with  $tp/fp$  the number of true/false positives (where a positive case denotes a phishing email), and with  $tn/fn$  the number of true/false negatives (where a negative case denotes a legitimate email), confusion matrix analysis allows the evaluation of specific metrics such as the following ones:

- Accuracy indicates the proportion of correct classifications ( $tp+tn$ ) to the total number of observations ( $tp+tn+fp+fn$ ). In other words, out of all the emails the model analyzed, accuracy estimates how many it classified correctly.
- Precision measures the proportion of correctly positive identifications ( $tp$ ) related to all positive



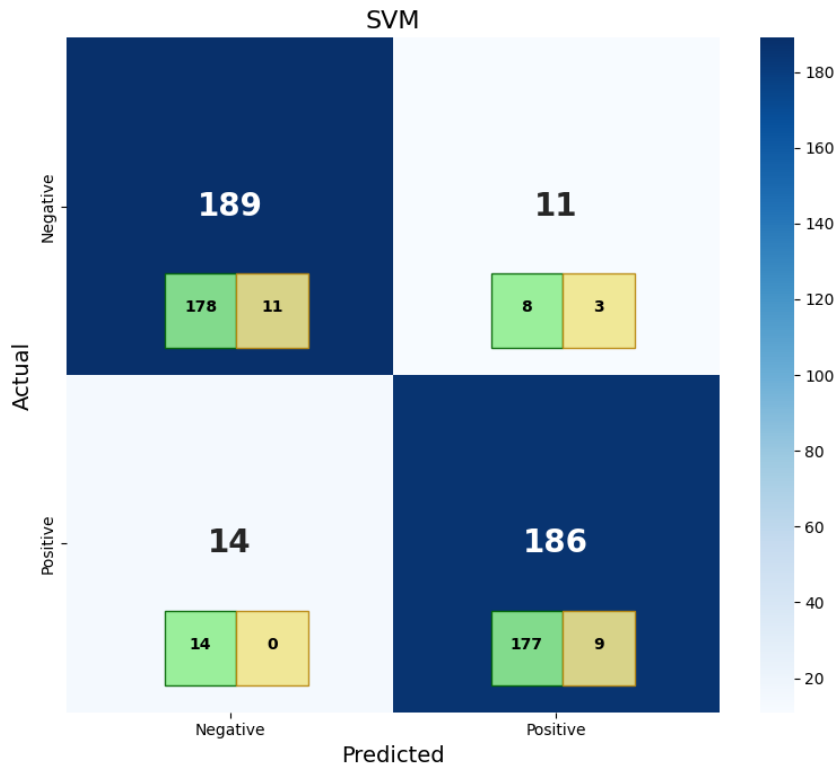
**Figure 3:** Performance RF classification: confusion matrix and comparison with LLM (Green: Gemini estimates correctly – Yellow: Gemini estimates incorrectly).

identifications generated by the classifier ( $tp+fp$ ). In our context, it quantifies the probability that an email classified as phishing is actually phishing.

- Recall measures the proportion of correctly identified true positives ( $tp$ ) compared to all actual positives ( $tp+fn$ ). In our context, it quantifies the ability of the classifier to identify all phishing emails in the dataset.
- F1-Score: is a harmonic mean of precision ( $prec$ ) and recall ( $rec$ ), i.e.,  $F1\text{-Score} = 2 \cdot prec \cdot rec / (prec + rec)$ ; it provides a balance between these two complementary metrics.

The confusion matrices are illustrated in Figures 3, 4, and 5. In particular, the classical representation of the confusion matrix has been enriched by including the results of the classification performed by Gemini. Hence, each cell of the matrix has been subdivided according to the concordance/discordance between Gemini and the classifier. This subdivision allows four main scenarios to be identified: both systems classify correctly (e.g., RF classifies correctly 187 legitimate emails and Gemini agrees for 174 of them, while RF classifies correctly 181 phishing emails and Gemini agrees for 173 of them); the classifier is correct while Gemini is not (with respect to the example above, Gemini incorrectly disagrees with RF for 13 legitimate emails and for 8 phishing emails); both systems make the same error (e.g., RF suffers from 13  $fp$ , one of which is a  $fp$  also in the case of Gemini, and from 19  $fn$ , one of which is a  $fn$  also in the case of Gemini); the classifier is incorrect while Gemini is correct (with respect to the example above, Gemini corrects 12 out of 13  $fp$  of RF and 18 out of 19  $fn$  of RF – see also the example of Figure 6, which is related to a XGBoost  $fp$ ). This integrated view allows us to evaluate the potential contribution of LLMs as complementary tools to traditional machine learning methods in classifying phishing emails.

First, let us comment on the performance of the three classification models, the results of which are reported in Table 1. All metrics exceed 90%. The SVM and XGBoost models both achieve the highest accuracy and F1-score, demonstrating an optimal balance between precision and recall. Both models outperform the RF classifier, which nevertheless guarantees good overall performance. The XGBoost model has the highest precision of all the classifiers, indicating a lower tendency to generate false



**Figure 4:** Performance SVM classification: confusion matrix and comparison with LLM (Green: Gemini estimates correctly – Yellow: Gemini estimates incorrectly).

positives, albeit with a slightly lower recall value than the SVM model.

Second, we evaluate the effect of using the LLM. In Table 2, we show the performance obtained by Gemini after the predictions of each classifier. The results emphasize that Gemini is not influenced by the output provided by each classifier; accuracy, recall, and F1-score are slightly higher than those of all the classifiers, while precision is lower in any case.

To complete the comparison between Gemini and the classifiers, we focus on the soundness of the evaluation of Gemini with respect to that of the classifiers. On the one hand, we consider how many times the LLM incorrectly disproves a true prediction by a classifier over the total number of true predictions of the same classifier. This metric is equal to 0.057 in the case of RF and 0.053 in the cases of SVM and XGBoost. These values somehow reflect the same differences observed when comparing the performance of the three classifiers. On the other hand, we consider how many times the LLM

**Table 1**

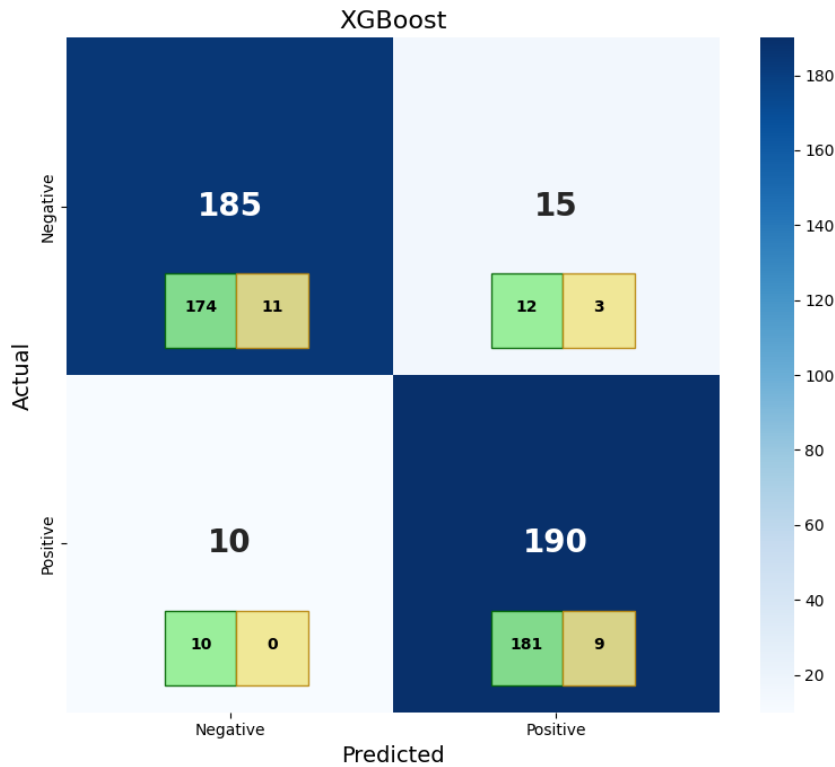
Performance of the classification models.

Classifier	Accuracy	Precision	Recall	F1-Score
Random Forest	0.920	0.933	0.905	0.919
SVM	0.938	0.944	0.930	0.937
XGBoost	0.938	0.949	0.925	0.937

**Table 2**

Performance of the LLM.

	Accuracy	Precision	Recall	F1-Score
Gemini + RF	0.942	0.932	0.955	0.943
Gemini + SVM	0.942	0.93	0.954	0.942
Gemini + XGBoost	0.942	0.93	0.954	0.942



**Figure 5:** Performance XGBoost classification: confusion matrix and comparison with LLM (Green: Gemini estimates correctly – Yellow: Gemini estimates incorrectly).

rightly corrects a false prediction by a classifier over the total number of false predictions of the same classifier. This metric is equal to 0.92 (resp., 0.95) in the case of RF *fp* (resp., RF *fn*). Interestingly, the LLM corrects SVM in 72.7% of *fp* cases and in 100% of *fn* cases, while it corrects XGBoost in 80% of *fp* cases and in 100% of *fn* cases.

## 5. Comparison with related work

Scientific research has produced numerous approaches to detecting phishing emails, ranging from techniques based on manual feature extraction and traditional classifiers to advanced deep learning methods and highly complex models. This section provides an overview of the significant works in the literature in this area, a key element of which is the dataset used for model development and validation. The datasets can vary significantly in size, composition, class balance, and acquisition methods. Most of the studies analyzed use public datasets such as SpamAssassin [30], Enron [31], APWG [32], Nazario [33].

Techniques based on manual feature extraction are an established methodology in phishing detection. In this context, Nimeh et al. [34] conducted a comparative analysis of several ML algorithms, including Logistic Regression (LR), Classification and Regression Trees (CART), Bayesian Additive Regression Trees (BART), Support Vector Machines (SVM), Random Forest (RF), and Neural Networks (NNet). In particular, compared to our framework, the implementation based on a dataset of 2889 emails and 43 extracted features showed the following performance: RF (precision 91.71%, recall 88.88%, F1-score 90.24%), SVM (precision 92.08%, recall 82.74%, F1-score 87.07%), NNet (precision 94.15%, recall 78.28%, F1-score 85.45%). Similarly, Olatunji [35] implements an SVM-based classifier with a set of 57 features, achieving 94.06% accuracy. Akinyelu et al. [13] construct a Random Forest classifier based on extracting 15 features from a dataset of 2000 emails derived from SpamAssassin and Nazario. The features, which include number of links, presence of HTML, suspicious words, message length, and mixed text/image

### Email Analysis

Email text

https://flutterflow.io/ We're back with our latest quarterly survey! This round, we're focusing on improving our community and learning resources. Your insights are invaluable, and we would love for you to participate in our survey to share your experiences and suggestions. 🙌 The survey closes on 8th May, 2024 12 PM, GMT. 📧 Link: https://flutterflow.typeform.com/q2-2024 Help us understand what works, what doesn't, and how we can develop more effective and supportive resources for all FlutterFlow users. 🏠 ===== \*\* FlutterFlow Logo (https://flutterflow.io) Build Different. You are receiving this email because you signed up on \*\* flutterflow.io (https://flutterflow.io) \*\* YouTube (https://www.youtube.com/channel/UC5LueiosDVlnA6yXE\_38i9Q) \*\* Twitter (http://www.twitter.com/flutterflow) \*\* LinkedIn (https://www.linkedin.com/company/flutterflow) \*\* Website (https://www.flutterflow.io) © 2024 FlutterFlow, Inc. Our mailing address: 883 N Shoreline Blvd, #B220, Mountain View, CA 94043 Want to change how you receive these emails? You can \*\* update your preferences (https://flutterflow.us2.list-manage.com/profile?u=\*\*\*\*\*&id=\*\*\*\*\*&e=\*\*\*\*\*&c=\*\*\*\*\*) or \*\* unsubscribe from this list (https://flutterflow.us2.list-manage.com/unsubscribe?u=\*\*\*\*\*&id=\*\*\*\*\*&t=b&e=\*\*\*\*\*&c=\*\*\*\*\*) .

Classifier

Random Forest

SVM

XGBoost

[Analyze](#)

### Analysis Result PHISHING

**Details:**

Classifier	XGBoost
Result	PHISHING

**Feedback:**

This prediction is correct?

Yes
No

**Advanced AI analytics:**

Request detailed email analysis using artificial intelligence

[🔍 Analyze with AI](#)

#### AI analysis

## Analysis of Email Classification

**\*\*1. Agreement with Initial Classification:\*\***

I **\*\*disagree\*\*** with the initial XGBoost classification of PHISHING. While XGBoost is a powerful classifier, it can produce false positives. My analysis suggests this email is **\*\*LEGITIMATE\*\***. The content and URLs point towards a genuine survey from FlutterFlow, a platform for building apps. The presence of "spam words" flagged by technical features might be due to marketing language inherent in survey invitations, not necessarily malicious intent.

**\*\*2. Suspicious Elements/Phishing Indicators:\*\***

There are **\*\*no significant suspicious elements\*\*** strongly indicating phishing. The email lacks typical phishing tactics like urgent threats, requests for sensitive information (passwords, financial details), or generic greetings. The survey request is reasonable for a platform seeking user feedback. The only potential flag is the initial classifier result, which warrants deeper investigation, but doesn't stand alone as strong evidence.

**\*\*3. URL Analysis:\*\***

The URLs are **\*\*legitimate and consistent\*\*** with FlutterFlow's online presence.

- `flutterflow.io`: Main website domain - legitimate.
- `flutterflow.typeform.com`: Uses Typeform, a reputable survey platform, under FlutterFlow subdomain - legitimate for surveys.
- Social media links (YouTube, Twitter, LinkedIn): Standard and correct domains - legitimate.
- `flutterflow.us2.list-manage.com`: This domain belongs to Mailchimp, a common email marketing service, and the subdomain `flutterflow` suggests it's used by FlutterFlow for legitimate email communications.
- `mcusercontent.com`: Mailchimp's content delivery network - legitimate for images in emails from Mailchimp users.

All URLs resolve to relevant services and domains associated with FlutterFlow or reputable third-party services they might utilize.

**\*\*4. Considerations on Technical Features:\*\***

- **\*\*Presence of URLs & No IP/Non-ASCII URLs:\*\*** Normal for legitimate emails.
- **\*\*Presence of typical spam words:\*\*** While flagged, this could be a false positive. Marketing emails, especially surveys, often contain words that might be generically flagged as "spam" but are contextually appropriate.
- **\*\*Sentiment Score (Positive):\*\*** Consistent with a legitimate survey invitation, indicating a positive and helpful tone rather than aggressive or threatening, which is typical of phishing.

**\*\*5. Final Conclusion:\*\***

I **classify this email as \*\*LEGITIMATE\*\***. Despite the initial PHISHING classification from the XGBoost model and the presence of "typical spam words" flag, a comprehensive analysis of the email content, URLs, and technical features indicates a genuine survey invitation from FlutterFlow. The use of reputable platforms like Typeform and Mailchimp, consistent domain names, and the absence of typical phishing indicators strongly support its legitimacy.

Figure 6: LLM analysis of a false positive generated by the XGBoost classifier.

content, were then reduced to 8 by Information Gain, allowing the following performance to be achieved through 10-fold cross-validation: 99.7% accuracy, 99.47% precision, and 97.50% recall. Similarly, Adewumi et al. [36] build a hybrid classifier combining the Firefly Algorithm (FFA) for parameter optimization with SVM support for ranking. This method is applied to a dataset of 4000 emails (3500 legitimate from SpamAssassin, 500 phishing from Nazario). The implementation, based on the selection of 8 out of 16 initial features, achieved remarkable results: accuracy of 99.98%, precision of 99.94%, recall of 99.92%, and F1-score of 99.93%. In particular, the hybrid FFA\_SVM methodology achieves superior results compared to standard SVM. Anirudh et al. [37] present an ensemble model integrating SVM and XGBoost with a final classifier based on logistic regression. The model, trained on a dataset of 18650 emails (11322 legitimate and 7328 phishing), achieved an accuracy of 97.92%, a precision of 98.43%, a recall of 98.22% and an F1-score of 98.32%.

The deep learning paradigm has introduced innovative methods that overcome the need for manual feature extraction. In this context, Atawneh and Aljehani [38] proposed a model based on Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM), Recurrent Neural Networks (RNNs), and Bidirectional Encoder Representations of Transformers (BERT). By integrating BERT and LSTM, an accuracy of 99.61% was achieved. An interesting application of CNNs is described in [39], where the authors developed a classifier for phishing URLs based on the conversion of feature vectors into images, achieving an accuracy of 86.5% on a dataset of 1353 real URLs classified into three categories: legitimate, suspicious, and phishing. A more sophisticated framework is proposed by Fang et al. [12], who developed a model called THEMIS, based on Recurrent Convolutional Neural Networks (RCNN), which is capable of simultaneously modeling the email header, body, character level, and word level, allowing in-depth analysis of the email. The model, evaluated on the IWSPA-AP [40] dataset consisting of 8780 emails, achieved high performance: accuracy of 99.85%, recall of 99.00%, precision of 99.66%, F1-score of 99.33%.

Another methodological advance is the Graph Convolutional Network (GCN) implementation proposed by Alhogail et al. [41]. The model, applied to a dataset of 3685 phishing and 4894 legitimate emails, uses a graph representation in which nodes correspond to emails and words, while weighted arcs represent the presence of words in the emails. This representation makes it possible to compute a matrix of features used to train the model, which achieved an accuracy of 98%, a precision of 98.5% and an F1-score of 98.5%. Doshi et al. [42] propose a two-layer architecture for both phishing and spam detection by addressing the data imbalance problem. The model, evaluated on a dataset of 8218 emails, integrates Artificial Neural Networks (ANN), RNNs, and CNNs, and achieves an accuracy of 99.51%, a recall of 99.68%, a precision of 99.5%, and an F1-score of 99.52%.

Of particular interest is the recent approach proposed by Nguyen et al. [43], based on a multi-agent system of LLMs that simulates a debate to determine the potential fraudulent nature of an email. The methodology implements two LLM agents that present arguments for or against the classification of phishing, and a third agent that judges the final verdict based on the quality of the arguments presented. The framework, evaluated on several datasets with a total of 12798 emails, showed that a heterogeneous configuration (GPT-4 as judge flanked by GPT-4 and LLaMA-2 agents) achieves a good performance with an accuracy of more than 98% on all the analyzed datasets. Similarly, Desolda et al. [2] present APOLLO, a GPT-4o-based system for detecting phishing emails and generating explanations of the danger of certain emails. The tool was evaluated on a dataset of 4000 emails and achieved an accuracy of 97%.

## 6. Conclusions

The pervasiveness of phishing attacks, combined with their continuous evolution, requires developing increasingly sophisticated and effective detection tools. In this work, we proposed an integrated phishing email detection system based on the combination of NLP techniques, ML algorithms, and an LLM. Experimental results confirmed the effectiveness of the proposed approach. In particular, the integration of ML and LLM techniques allows for a more detailed classification and feedback, while

also highlighting the potential differences between these models. Hence, it would be meaningful to address the presence, if any, of specific patterns that characterize the cases in which the different models disagree.

As future work, we plan to use the feedback data to build historical statistics of the system's accuracy. It is also interesting to investigate how the feedback provided by the different models might be used to affect the accuracy of the ML classifiers or to enrich the prompts for the LLM. While the experimental setup did not reveal a significant overhead due to the use of the detection system at run-time, it is still worth evaluating the performance of implementations for mobile devices and federated learning scenarios. Moreover, we plan to expand the dataset to improve the system's performance further and test its robustness on more complex and varied cases. Sector specialization represents another promising direction as training models on specific categories of phishing emails, contextualized by particular industry sectors (banking, healthcare, education), could significantly improve performance. This specialization would allow classifiers to be refined to detect fraudulent patterns characteristic of each domain, more effectively excluding irrelevant patterns and reducing false positives. We also plan to apply the same pipeline of different techniques to assess the secure behavior of software in smart environments [44, 45, 46]. Finally, the integration with CNN-based models [47] and visual analytics techniques to detect image-based phishing could be explored and adapted for other domains, such as deepfake detection or classification of malicious content on social media.

## Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

## References

- [1] S. Salloum, T. Gaber, S. Vadera, K. Shaalan, Phishing email detection using natural language processing techniques: A literature survey, *Procedia Computer Science* 189 (2021) 19–28. doi:10.1016/j.procs.2021.05.077.
- [2] G. Desolda, F. Greco, L. Vigano, Apollo: A gpt-based tool to detect phishing emails and generate explanations that warn users, *Proc. ACM Hum.-Comput. Interact.* 9 (2025). doi:10.1145/3733049.
- [3] S. Salloum, T. Gaber, S. Vadera, K. Shaalan, A systematic literature review on phishing email detection using natural language processing techniques, *IEEE Access* 10 (2022) 65703–65727. doi:10.1109/ACCESS.2022.3183083.
- [4] Verizon Business, Data breach investigations report, <https://www.verizon.com/business/resources/reports/dbir/>, 2025.
- [5] 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. doi:10.1109/SURV.2013.030713.00020.
- [6] A. Oest, Y. Safaei, P. Zhang, B. Wardman, K. Tyers, Y. Shoshitaishvili, A. Doupé, PhishTime: Continuous longitudinal measurement of the effectiveness of anti-phishing blacklists, in: 29th USENIX Security Symposium (USENIX Security 20), USENIX Association, 2020, pp. 379–396. URL: <https://www.usenix.org/conference/usenixsecurity20/presentation/oest-phishtime>.
- [7] T. Gangavarapu, C. Jaidhar, B. Chanduka, Applicability of machine learning in spam and phishing email filtering: review and approaches, *Artificial Intelligence Review* 53 (2020). doi:10.1007/s10462-020-09814-9.
- [8] Microsoft Security Team, Microsoft digital defense report 2023, <https://www.microsoft.com/en-us/security/security-insider/microsoft-digital-defense-report-2023#modal-dialog>, 2023.
- [9] Y. Kang, Z. Cai, C.-W. Tan, Q. Huang, H. Liu, Natural language processing (NLP) in management research: A literature review, *Journal of Management Analytics* 7 (2020) 139–172. doi:10.1080/23270012.2020.1756939.

- [10] V. Shahrivari, M. M. Darabi, M. Izadi, Phishing detection using machine learning techniques, 2020. doi:10.48550/arXiv.2009.11116.
- [11] O. K. Sahingoz, E. Buber, O. Demir, B. Diri, Machine learning based phishing detection from URLs, *Expert Systems with Applications* 117 (2019) 345–357. doi:<https://doi.org/10.1016/j.eswa.2018.09.029>.
- [12] Y. Fang, C. Zhang, C. Huang, L. Liu, Y. Yang, Phishing email detection using improved RCNN model with multilevel vectors and attention mechanism, *IEEE Access* 7 (2019) 56329–56340. doi:10.1109/ACCESS.2019.2913705.
- [13] A. A. Akinyelu, A. O. Adewumi, Classification of phishing email using random forest machine learning technique, *Journal of Applied Mathematics* 2014 (2014). doi:10.1155/2014/425731.
- [14] V. Muppavarapu, A. Rajendran, S. Vasudevan, Phishing detection using rdf and random forests, *The International Arab Journal of Information Technology (IAJIT)* 15 (2018) 817–824. doi:10.34028/iajit.
- [15] T. Kim, N. Park, J. Hong, S.-W. Kim, Phishing url detection: A network-based approach robust to evasion, in: *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS'22*, ACM, New York, NY, USA, 2022, p. 1769–1782. doi:10.1145/3548606.3560615.
- [16] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: J. Burstein, C. Doran, T. Solorio (Eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. doi:10.18653/v1/N19-1423.
- [17] L. Breiman, Random forests, *Machine Learning* 45 (2001) 5–32. doi:10.1023/A:1010933404324.
- [18] C. Cortes, V. Vapnik, Support-vector networks, *Machine Learning* 20 (1995) 273–297. doi:10.1007/BF00994018.
- [19] P. Hajek, M. Z. Abedin, U. Sivarajah, Fraud detection in mobile payment systems using an XGBoost-based framework, *Information Systems Frontiers* 25 (2023) 1985–2003. doi:10.1007/s10796-022-10346-6.
- [20] T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'16*, ACM, 2016, p. 785–794. doi:10.1145/2939672.2939785.
- [21] D. Rosati, EmailAnalyzer, <https://github.com/daniloros/EmailAnalyzer>, 2025.
- [22] University of Waikato, Use weka in your java code, [https://waikato.github.io/weka-wiki/use\\_weka\\_in\\_your\\_java\\_code/](https://waikato.github.io/weka-wiki/use_weka_in_your_java_code/), 2026.
- [23] XGBoost Developers, XGBoost documentation – release 3.0.0, [https://xgboost.readthedocs.io/en/release\\_3.0.0/](https://xgboost.readthedocs.io/en/release_3.0.0/), 2022.
- [24] Google Cloud, Natural language API documentation, <https://cloud.google.com/natural-language/docs/>, 2026.
- [25] Google, Gemini API models, <http://ai.google.dev/gemini-api/>, 2026.
- [26] Spring Team, Spring boot project, <https://spring.io/projects/spring-boot>, 2026.
- [27] Eclipse Foundation, Jakarta mail specification, <https://jakarta.ee/specifications/mail/>, 2026.
- [28] J. Hedley, jsoup API documentation, <https://jsoup.org/apidocs/>, 2026.
- [29] S. Chakraborty, Phishing email detection, 2023. doi:10.34740/KAGGLE/DSV/6090437, kaggle.
- [30] The Apache Software Foundation, Apache spamassassin: Welcome to spamassassin, <https://spamassassin.apache.org/>, 2025.
- [31] Carnegie Mellon University, Enron email dataset, <https://www.cs.cmu.edu/~enron/>, 2015.
- [32] Anti-Phishing Working Group, APWG - Unifying the global response to cybercrime, <https://apwg.org/>, 2026.
- [33] N. A. Alam, Phishing email dataset, 2024. doi:10.34740/KAGGLE/DS/5074342, kaggle.
- [34] S. Abu-Nimeh, D. Nappa, X. Wang, S. Nair, A comparison of machine learning techniques for phishing detection, in: *Proceedings of the Anti-Phishing Working Groups 2nd Annual ECrime Researchers Summit, eCrime'07*, ACM, New York, NY, USA, 2007, p. 60–69. doi:10.1145/1299015.1299021.

- [35] S. O. Olatunji, Improved email spam detection model based on support vector machines, *Neural Computing and Applications* 31 (2019) 691–699. doi:10.1007/s00521-017-3100-y.
- [36] A. Adewumi, A. Akinyelu, A hybrid firefly and support vector machine classifier for phishing email detection, *Kybernetes* 45 (2016) 977–994. doi:10.1108/K-07-2014-0129.
- [37] S. Anirudh, P. Radha Nishant, S. Baitha, K. Dinesh Kumar, An ensemble classification model for phishing mail detection, *Procedia Computer Science* 233 (2024) 970–978. doi:https://doi.org/10.1016/j.procs.2024.03.286.
- [38] S. Atawneh, H. Aljehani, Phishing email detection model using deep learning, *Electronics* 12 (2023). doi:10.3390/electronics12204261.
- [39] C. McGinley, S. A. S. Monroy, Convolutional neural network optimization for phishing email classification, in: *2021 IEEE International Conference on Big Data (Big Data)*, 2021, pp. 5609–5613. doi:10.1109/BigData52589.2021.9671531.
- [40] IWSPA Anti-Phishing Working Group, IWSPA anti-phishing shared task (IWSPA-AP), <https://dasavisha.github.io/IWSPA-sharedtask/>, 2018.
- [41] A. Alhogail, A. Alsabih, Applying machine learning and natural language processing to detect phishing email, *Computers & Security* 110 (2021) 102414. doi:10.1016/j.cose.2021.102414.
- [42] J. Doshi, K. Parmar, R. Sanghavi, N. Shekokar, A comprehensive dual-layer architecture for phishing and spam email detection, *Computers & Security* 133 (2023) 103378. doi:10.1016/j.cose.2023.103378.
- [43] N. T. Vy Nguyen, F. D. Childress, Y. Yin, Debate-driven multi-agent llms for phishing email detection, in: *2025 13th International Symposium on Digital Forensics and Security (ISDFS)*, IEEE, 2025, p. 1–5. doi:10.1109/isdfs65363.2025.11012014.
- [44] A. Aldini, L. Ardito, G. M. Bianco, M. Valsesia, Lich: Enhancing iot supply chain security through automated firmware analysis, in: *2025 21st International Conference on Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT)*, 2025, pp. 747–754. doi:10.1109/DCOSS-IoT65416.2025.00114.
- [45] A. Aldini, L. Ardito, G. M. Bianco, M. Valsesia, A rust library for behaviors assessment in software certification, *IEEE Internet of Things Journal* 12 (2025) 42303–42316. doi:10.1109/JIOT.2025.3592839.
- [46] M. Rasori, P. Mori, A. Saracino, A. Aldini, Exploiting usage control for implementation and enforcement of security by contract, *Internet of Things* 33 (2025) 101697. doi:10.1016/j.iot.2025.101697.
- [47] A. Aldini, T. Petrelli, Image-based detection and classification of android malware through CNN models, in: *Proceedings of the 19th International Conference on Availability, Reliability and Security, ARES 2024*, ACM, 2024, pp. 61:1–61:11. doi:10.1145/3664476.3670441.