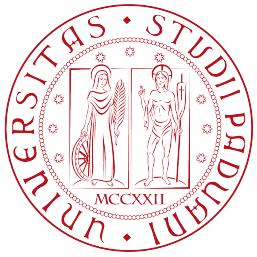


UNIVERSITY OF PADUA

PH.D. SCHOOL IN BRAIN, MIND AND COMPUTER SCIENCE
Curriculum in Computer Science and Innovation for Societal Challenges
XXXVI Cycle

Machine Learning for Phishing Website Detection

Candidate:
Ying YUAN

Supervisor:
Prof. Mauro CONTI
University of Padua

Co-Supervisor:
Prof. Anna Spagnolli
University of Padua

Acknowledgements

First of all, I would like to give my heartfelt thanks to my supervisor, Prof. Mauro Conti, for his invaluable instruction, inspiration and patience. He introduced me to many nice professors and gave me a chance to work with them, including Prof. Giovanni Apruzzese and Prof. Gang Wang. It's my great honour to join SPRITZ group and meet wonderful colleagues here, including Luca Pajola, Federico Turrin, Stefano Cecconello, Pier Paolo Tricomi, Alessandro Visintin and Jiaxin Li. Also, I would like to express my gratitude to everyone at 732 for the happy time we had together. I had an unforgettable time in Italy. In addition, I must thank my parents, my younger brother and my friend Yalan Wang, I can not live abroad without their support and inspiration. Last but not least, a special thanks to Prof. Giovanni Apruzzese, for his kindness, illuminating guidance and profound knowledge. It was quite inspiring and exciting to work with him. I am really grateful to all those who devote much time to reading this thesis and give me much advice, which will benefit me in my later studies.

Abstract

Phishing attacks are on the rise and phishing websites are everywhere, denoting the brittleness of security mechanisms reliant on blocklists. Prior work proposed enhancing Phishing Website Detectors (PWD) to mitigate this threat with data-driven techniques powered by Machine Learning (ML). The main advantage of ML models is their intrinsic ability of noticing weak patterns in the data that are overlooked by a human, and then leveraging such patterns to devise 'flexible' detectors that can counter even adaptive attackers.

This dissertation addresses three significant aspects arising from the interaction between machine learning and phishing website detection: (i) Adversarial attack for machine learning-based phishing website detection (ML-PWD), (ii) User perceptions of Phishing webpages, and (iii) Phishing website detection in multi-language environment (i.e., Chinese and Western)

The first part presents the security of ML-based phishing website detection. Existing literature on adversarial Machine Learning (ML) focuses either on showing attacks that break every ML model, or defenses that withstand most attacks. Unfortunately, little consideration is given to the actual cost of the attack or the defense. We formalize the "evasion-space" in which an adversarial perturbation can be introduced to fool a ML-PWD and propose a realistic threat model describing evasion attacks against ML-PWD that are cheap to stage. Our contribution paves the way for a much-needed re-assessment of adversarial attacks against ML systems for cybersecurity. The second part of the dissertation presents a study to understand user perceptions of phishing and adversarial phishing webpages. Adversarial phishing webpages containing perturbations can easily fool ML-based PWD, but it remains uncertain whether these perturbations could equally deceive the real target-end users. Our study indicates adversarial phishing webpages containing typos are more likely to be perceived by users. The third - and last - part of the dissertation reveals the gap between Chinese and Western ML-based PWD, aiming to urge that future work in PWD should take into account the applicability of multilingual environments and pave the way for PWD systems that can protect users having different backgrounds.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Phishing is the topmost form of cybercrime according to the FBI's Internet crime report [10]. In the second quarter of 2022, the Anti-Phishing Working Group reported over 1M phishing attacks—the worst quarter ever observed [24]. In this context, phishing *websites* represent one of the most common vectors employed by attackers, who aim to reach their goals by tricking their victims via apparently legitimate websites [28]. In the first half of 2022, over 200k phishing websites were generated every month [213]—showing that a universal solution to this threat has yet to be found.

The subject of Phishing Website Detection (PWD) is well-studied both in academia and industry. Lots of anti-phishing schemes have been proposed, either "human" centered, such as phishing education (e.g., [148, 157]); or "machine" centered, such as automated detection methods (e.g., [134, 222]). This paper focuses on the latter, which does not require any prior knowledge of phishing by potential victims.

Automated PWD can leverage two detection approaches (or a combination thereof), based on either signature (in the form of "blocklists" [215]), or on data-driven heuristics (e.g., [50, 142, 144]). The former is widely used in browsers; for instance, Google Safe Browsing [32] relies on a constantly updated blocklist which is checked before opening any website, thereby raising an alert if the visited URL (or part of it) is included in such a blocklist. Despite being very precise (i.e., low rates of false positives), blocklist-based PWD cannot detect 'novel' phishing websites [198, 199]. To overcome this limitation, some advanced PWD leverage data-driven methods in the domain of Machine Learning (ML): the intuition is to analyze some "features" of a website (extracted from, e.g., its URL or even the underlying HTML [276]) to discriminate benign from malicious webpages. ML-based PWD (ML-PWD) are capable of detecting phishing webpages not included in any blocklist [247], but at the expense of a superior (but still acceptable [55]) rate of false alarms.

The cornerstone of ML is having "machines that automatically learn from experience" [150], and such experience comes in the form of *data*. ML models can notice weak patterns in the data that are overlooked by a human with the help of this intrinsic ability, and then leverage such patterns to devise 'flexible' detectors that can counter even adaptive attackers. As a matter of fact, Tian et al. [247] show that a ML model based on Random Forest (RF) is effective even against "squatting" phishing websites—while retaining a low-rate of false alarms (only 3%). Moreover, acquiring suitable data (i.e., recent and labelled) for ML-PWD is not difficult—compared to other cyber-detection problems for which ML has been proposed [62].

Such advantages have been successfully leveraged by many research efforts (e.g., [196, 241]). Existing ML-empowered PWD can leverage different types of *information* (i.e., *features*) to perform their detection. Such information can pertain either to a website's *URL* [255] or to its *representation*, e.g., by analyzing the actual image

of a webpage as rendered by the browser [131], or by inspecting the HTML [144]. For example, Mohammad et al. [188] observed that phishing websites usually have long URLs; and often contain many 'external' links (pointing to, e.g., the legitimate 'branded' website, or the server for storing the phished data), which can be inferred from the underlying HTML. Although some works use only URL-related features (e.g., [85]) – which can also be integrated into phishing *email* filters (e.g., [128]) – more recent proposals use combinations of features (e.g., [97, 252]); potentially, such features can be derived by querying third-party services (e.g., DNS servers [143]).

The cost-effectiveness of ML-PWD increased their adoption: even commercial browsers (e.g., Google Chrome [171]) integrate ML models in their phishing filters (which can be further enhanced via customized add-ons [242]); moreover, ML-PWD can also be deployed in corporate SIEM [136].

1.1 Research Motivation and Contribution

This thesis mainly investigates issues in Machine learning-based Phishing website detection, focusing on three major aspects.

1. *Adversarial attack for machine learning-based phishing website detection*: solutions aiming to estimate the actual threat posed by adversarial attacks in the field of Phishing website detection. Chapter 2 formalized the "evasion-space" in which an adversarial perturbation can be introduced to fool the ML-PWD, and proposed a realistic threat model describing evasion attacks against ML-PWD that is cheap to stage. All attacks occur in a single space. Chapter 3 considers a "stronger" attacker that applies multiple perturbations in mixed evasion spaces.

2. *User perceptions of Phishing webpages*: aiming to estimate user perceptions on phishing webpages. Chapter 4 focus on answering whether adversarial phishing websites equally deceive users as deceiving machine learning models, and elucidate users' awareness of phishing websites.

3. *Phishing website detection in multi-language environment*: aiming to evaluate the "cross-language" effectiveness of state-of-the-art PWD and reveal the difference between Western and Chinese phishing websites (i.e., phonetic and hieroglyphics language-based phishing websites). Chapter 5 presents Chphish, a study aiming to elucidate and bridge the gap between Western and Chinese phishing website detection.

In this dissertation, some passages have been quoted verbatim, and some figures have been reused from the work [59], coauthored by the author of the thesis.

SpacePhish: The Evasion-space of Adversarial Attacks against Phishing Website Detectors using Machine Learning

Existing literature on adversarial Machine Learning focuses either on showing attacks that break every ML model, or defenses that withstand most attacks. Unfortunately, little consideration is given to the actual *cost* of the attack or the defense. Moreover, adversarial samples are often crafted in the "feature-space", making the corresponding evaluations of questionable value. Simply put, the current situation does not allow to estimate the actual threat posed by adversarial attacks, leading to a lack of secure ML systems.

Contribution In Chapter 2, We aim to clarify such confusion in this paper. By considering the application of ML for Phishing Website Detection, we formalize the "evasion-space" in which an adversarial perturbation can be introduced to fool a ML-PWD—demonstrating that even perturbations in the "feature-space" are useful. Then, we propose a realistic threat model describing evasion attacks against ML-PWD that are cheap to stage, and hence intrinsically more attractive for real phishers. Finally, we perform the first statistically validated assessment of state-of-the-art ML-PWD against 12 evasion attacks. Our evaluation shows (i) the true efficacy of evasion attempts that are more likely to occur; and (ii) the impact of perturbations crafted in different evasion-spaces. Our realistic evasion attempts induce a statistically significant degradation (3–10% at $p < 0.05$), and their cheap cost makes them a subtle threat. Notably, however, some ML-PWD are immune to our most realistic attacks (p=0.22). Our contribution paves the way for a much needed re-assessment of adversarial attacks against ML systems for cybersecurity.

Multi-SpacePhish: Extending the Evasion-space of Adversarial Attacks against Phishing Website Detectors using Machine Learning

Research papers intrinsically impair the development of secure ML systems, because the aim is often to "outperform the state-of-the-art". In adversarial ML, this leads to papers that either showcase devastating attacks stemming from extremely powerful adversaries (i.e., white-box [237]); or vice versa, i.e., show that even oblivious attackers can thwart ML systems [207]. However, real 'adaptive' attackers (i.e., those that ML methods should be protected against) do not conform to these two extremes. Indeed, having complete knowledge of the target system requires a huge resource investment (especially if such system is devoted to cybersecurity), which may be better spent elsewhere; conversely, it is unlikely that opponents will launch attacks while knowing nothing of the defender. Hence, to provide *valuable* research, efforts on adversarial ML should start focusing on the gray area within these two extremes—which implicitly are more likely to occur [56]. In the context of ML-PWD, our paper is a first step in this direction: despite being devastating, existing evasion attempts are costly to launch—even in black-box settings.

Contribution In Chapter 3, we propose and empirically evaluate the intriguing case wherein an attacker introduces perturbations in multiple evasion spaces *simultaneously*. This work extended the single space attack in [59] to 'deeper' multiple spaces. These new results show that applying perturbations in the problem- and feature-space *at the same time* can lead to a significant decrease in the detection rate from 0.95 to 0.

Understanding User Perceptions of Adversarial Phishing Websites

Machine learning based phishing website detectors (ML-PWD) are a critical part of today's anti-phishing solutions in operation. Unfortunately, ML-PWD are prone to adversarial evasions, evidenced by both academic studies and analyses of real-world adversarial phishing webpages. However, existing works mostly focused on assessing adversarial phishing webpages against ML-PWD, while neglecting a crucial aspect: investigating whether they can deceive the actual target of phishing—the end users.

Contribution In Chapter 4, we fill this gap by conducting two user studies (n=470) to examine how human users perceive adversarial phishing webpages, spanning both synthetically crafted ones (which we create by evading a state-of-the-art ML-PWD) as well as real adversarial webpages (taken from the wild Web) that bypassed a production-grade ML-PWD. Our findings confirm that adversarial phishing is a threat to both users and ML-PWD, since most adversarial phishing webpages have comparable effectiveness on users w.r.t. unperturbed ones. However, not all adversarial perturbations are equally effective. For example, those with added typos are significantly more noticeable to users, who tend to overlook perturbations of higher visual magnitude (such as replacing the background). We also show that users' self-reported frequency of visiting a brand's website has a statistically negative correlation with their phishing detection accuracy, which is likely caused by overconfidence.

ChinaPhish: Revealing, Assessing, and Bridging the Gap between Western and Chinese Phishing Website Detection

Despite existing ML-based phishing website detectors achieving promising results both in research and practice, they mostly focus on "western" websites, e.g., they consider websites in English, German, or Italian. In contrast, phishing websites targeting "eastern" countries, such as China, have been mostly neglected—despite phishing being rampant also on this side of the world.

The **motivation** of this study has its root in the fact that: *(i)* an increasing number of Western people now reside in China [16], and that *(ii)* an increasing number of Chinese people migrated to the West [11]. As such, it is important to scrutinize whether phishing website detectors can "transfer" between different regions: For instance, an English person can be protected if they live in the UK and only visit English websites—but what if such a person goes to China and starts visiting (also) Chinese websites? And, vice-versa, previously proposed Chinese PWD may be effective as long as they are integrated into browsers used in China—but what if a Chinese person goes abroad and starts visiting Western websites?

Contributions In Chapter 5, we scrutinize whether the current phishing website detectors can simultaneously work against Western and Chinese phishing websites. We first elucidate the differences between Western and Chinese websites, in terms of textual language and webpage structure—suggesting that existing PWD for "western" websites may not work on Chinese ones. Then, we empirically prove the existence of a gap between Chinese and Western PWD. Practically, we evaluate 61 commercial PWD and 89 ML-based PWD on three datasets containing thousands of websites of different languages—including a novel dataset for Chinese PWD, the *first* of its kind, which we publicly release. Our experiments reveal that PWD tailored for Western websites perform poorly when tested on Chinese websites, with F1-score dropping by 19%–47%, and vice-versa. The gap we identified is not acceptable today, given the increasing migratory waves from/to diverse areas of the World. Our takeaway is that future work on PWD should stop focusing only on Western websites, thereby paving the way for PWD systems that can protect users having different backgrounds.

1.1.1 Publications

This section summarizes manuscripts produced during my Ph.D. period and published or currently submitted in peer-reviewed journal and conferences. All manuscripts are listed in chronological order of acceptance and submission.

Journal Publication

1. Yuan, Y., Apruzzese, G., & Conti, M. (2023). Multi-SpacePhish: Extending the Evasion-space of Adversarial Attacks against Phishing Website Detectors using Machine Learning. *ACM Digital Threats: Research and Practice.* Submitted

Conference Publication

- Apruzzese, G., Conti, M., & Yuan, Y. (2022, December). SpacePhish: The Evasion-space of Adversarial Attacks against Phishing Website Detectors using Machine Learning. *In Proceedings of the 38th Annual Computer Security Applications Conference (ACSAC)* (pp. 171-185). **(LiveSHINE:A+; MA:A+)**[reusable badge]

- Yuan, Y., Apruzzese, G., & Conti, M. (2023), ChinaPhish: Revealing the Gap between Western and Chinese Phishing Website Detection. *IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE. (**MA:C**) Submitted

- Yuan, Y., Apruzzese, G., Hao, Q., Conti, M., & Wang, G. (2023), Understanding User Perceptions of Adversarial Phishing Websites. *ACM International World Wide Web Conference (The WWW)*. ACM. **(GGS A++/1, CORE:A++, LiveSHINE:A++, MA:A++)** Submitted

- Hao, Q., Diwan, N., Yuan, Y., Apruzzese, G., Conti, M., & Wang, G. (2023), It Doesn't Look Like Anything to Me: Using Diffusion Model to Subvert Visual Phishing Detectors. *The USENIX Security Symposium*. **(GGS: 1/A++; CORE: A++; LiveSHINE: A++; MA: A++)** Submitted

Part I

Adversarial attack for machine learning-based phishing website detection

Chapter 2

SpacePhish: The Evasion-space of Adversarial Attacks against Phishing Website Detectors using Machine Learning

After more than a decade of research [80] and thousands of papers [15], it is well-known that Machine Learning (ML) methods are vulnerable to adversarial attacks. Specifically, by introducing imperceptible perturbations (down to a single pixel or byte [57, 237]) in the input data, it is possible to compromise the predictions made by a ML model. Such vulnerability, however, is more dangerous in settings that implicitly assume the presence of adversaries. A cat will not try to fool a ML model. An attacker, in contrast, will actively try to evade a ML detector—the focus of this paper.

On the surface, the situation portrayed in research is vexing. The confirmed successes of ML [150] are leading to large-scale deployment of ML in production settings (e.g., [103, 219, 242]). At the same time, however, dozens of papers showcase adversarial attacks that can crack 'any' ML-based detector (e.g., [58, 171]). Although some papers propose countermeasures (e.g., [209]), they are quickly defeated (e.g., [89]), and typically decrease the baseline performance (e.g. [58, 104]). As a result, recent reports [115, 156] focusing on the integration of ML *in practice* reveal that: "I Never Thought About Securing My Machine Learning Systems" [82]. This is not surprising: if ML can be so easily broken, then why invest resources in increasing its security through –unreliable– defenses?

Sovereign entities (e.g., [8, 13]) are endorsing the development of "trustworthy" ML systems; yet, any enhancement should be economically justified. No system is foolproof (ML-based or not [87]), and guaranteeing protection against omnipotent attackers is an enticing but unattainable objective. In our case, a security system should increase the *cost* incurred by an attacker to achieve their goal [191]. Real attackers have a cost/benefit mindset [259]: they may try to evade a detector, but only if doing so yields positive returns. In reality, worst-case scenarios are an exception—not the norm.

This study is inspired by several recent works that pointed out some 'inconsistencies' in the adversarial attacks carried out by prior studies. Pierazzi et al. [214] observe that real attackers operate in the "problem-space", i.e., the perturbations they can introduce are subject to physical constraints. If such constraints are not met, and hence the perturbation is introduced in the "feature-space" (e.g., [195]), then there is a risk of generating an adversarial example that is not physically realizable [248]. Apruzzese et al. [56], however, highlight that even 'impossible' perturbations can be

applied, but *only if* the attacker has internal access to the data-processing pipeline of the target system. Nonetheless, Biggio and Roli suggest that ML security should focus on "anticipating the most likely threats" [80]. Only *after* proactively assessing the impact of such threats a suitable countermeasure can be developed—if required.

We aim to promote the development (and deployment) of secure ML systems. However, meeting Biggio and Roli's recommendation presents two tough challenges for research papers. First, it is necessary to devise a *realistic threat model* which portrays adversarial attacks that are not only physically realizable, but also economically viable. Devising such a threat model, however, requires a detailed security analysis of the *specific cyberthreat* addressed by the detector—while factoring the resources that attackers are willing to invest. Second, it is necessary to *evaluate the impact* of the attack by crafting the corresponding perturbations. Doing so is difficult if the threat model assumes an attacker operating in the problem-space, because such *perturbations must be applied on raw-data*, i.e., before any preprocessing occurs—which is hard to find.

In this paper, we tackle both of these challenges. In particular, we focus on ML-systems for Phishing Website Detection (PWD). Countering phishing – still a major threat today [28, 151] – is an endless struggle. Blocklists can be easily evaded [247], and to cope against adaptive attackers some detectors are equipped with ML (e.g. [242]). Yet, as shown by Liang et al. [171], even such ML-PWD can be "cracked" by oblivious attackers—if they invest enough effort to reverse engineer the entire ML-PWD. Indeed, we address ML-PWD because prior work (e.g., [74, 124, 165, 229]) assumed threat models that hardly resemble a real scenario. Phishing, by nature, is meant to be cheap [152] and most attempts end up in failure [200]. It is unlikely[1] that a phisher invests many resources *just to evade* ML-PWD: even if a website is not detected, the user may be 'hooked', but is not 'phished' yet. As a result, the state-of-the-art on adversarial ML for PWD is immature—from a pragmatic perspective.

Contribution and Organization. Let us explain how we aim to spearhead the security enhancements to ML-PWD. We begin by introducing the fundamentals concepts (PWD, ML, and adversarial ML) at the base of this study in §2.1, which also serves as a motivation. Then, we make the following four contributions.

- We *formalize the evasion-space* of adversarial attacks against ML-PWD (§2.2), rooted in exhaustive analyses of a generic ML-PWD. Such evasion-space explains 'where' a perturbation can be introduced to fool a ML-PWD. Our formalization highlights that even adversarial samples created by direct feature manipulation can be realistic, *validating all the attacks performed by past work*.

- By using our formalization as a stepping stone, we propose a *realistic threat model* for evasion attacks against ML-PWD (§2.3). Our threat model is grounded on detailed security considerations from the viewpoint of a typical phisher, who is confined in the 'website-space'. Nevertheless, our model can be relaxed by assuming attackers with greater capabilities (which require a higher cost).

- We combine and practically demonstrate the two previous contributions (§2.4). We perform an extensive, reproducible, and statistically validated *evaluation of adversarial attacks* against state-of-the-art ML-PWD. By using diverse datasets,

[1] It is unlikely, but *not impossible*. Hence, as recommended by Arp et al [70], it is positive that such cases have also been studied by prior work.

ML algorithms and features, we develop 18 ML-PWD, each of which is assessed against 12 different evasion attacks built upon our threat model.

- By analyzing the results of our evaluation (§2.5): (i) we *show the impact of attacks that are very likely to occur* against both baseline and adversarially robust ML-PWD; and (ii) we are the first to *fairly compare the effectiveness* of evasion attacks in the problem-space with those in the feature-space.

Our results highlight that more realistic attacks are not as disruptive as claimed by past works (§2.6), but their low-cost makes them a threat that induces statistically significant degradations. Finally, our evaluation serves as a 'benchmark' for future studies: we provide the complete results and source-code in a dedicated website: `https://spacephish.github.io`.

2.1 Background and Motivation

This study lies at the intersection of Phishing Website Detection (PWD) and Machine Learning (ML) security. To set-up the stage for our contribution and motivate its necessity, we first summarize PWD (§2.1.1), and then explain the role of ML in PWD (§2.1.2). Finally, we provide an overview of the adversarial ML domain (§2.1.3).

2.1.1 Phishing Website Detection

Although having been studied for nearly two decades [153], phishing attacks are still a rampant menace [151]: according to the FBI [7], the number of reported phishing attempts has increased by 900% from 2018 to 2020 (26k up to 240k). Aside from the well-known risks to single users (e.g., fraud, credential theft [127]), phishing is still one of the most common vectors to penetrate an organization's perimeter. Intuitively, the best countermeasure to phishing is its prevention through proper *education* [264]. Despite recent positive trends, however, such education is far from comprehensive: the latest "State of the Phish" report [28] states that more than 33% of companies do not have any training program for their employees, and more than 50% only evaluate such education through simulations. As a result, there is still a need of IT solutions that mitigate the phishing threat by its early *detection*. In our case, this entails identifying a phishing website before a user lands on its webpage, therefore defusing the risk of falling victim to a phishing attack. We provide in Fig. 2.1 an exemplary architecture of a Phishing Website Detector (PWD).

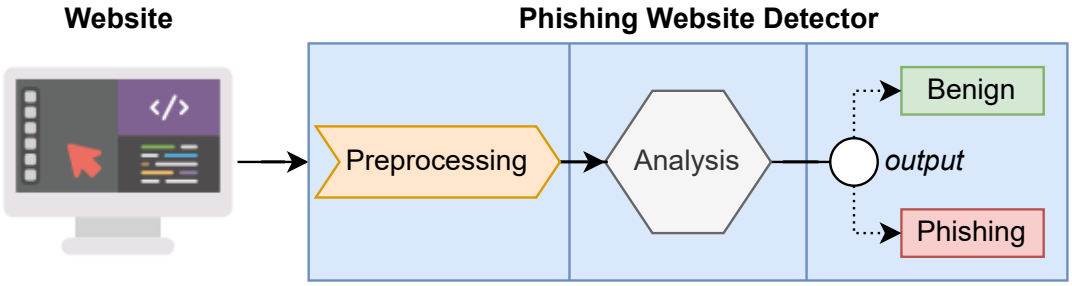


FIGURE 2.1: Exemplary PWD. After preliminary preprocessing, a website is analyzed by a detector to determine its legitimacy.

Despite extensive efforts, PWD remains an open issue. This is due to the intrinsic limitations of the most common detection approaches reliant on *blocklisting* (e.g., [199, 215]). Such techniques have been improved and nowadays they even

involve automatic updates with recent feeds (e.g., PhishTank [26]). However, blocklists are a double-edged sword: on the good side, they are very precise and are hence favored due to the low rate of false alarms; on the bad side, they are only effective against known phishing websites [45]. The latter is a problem: expert attackers are aware of blocklists and hence move their phishing 'hooks' from site to site, bypassing most PWD. As shown by Tian et al. [247], such strategies can elude over 90% of popular blocklists for more than one month. To counter such *adaptive* attackers, much attention has been given to data-driven detection schemes—including those within the Machine Learning (ML) paradigm [242]. Indeed, ML allows to greatly enhance the detection capabilities of PWD. Let us explain why.

2.1.2 Machine Learning for PWD

The cornerstone of ML is having "machines that automatically learn from experience" [150], and such experience comes in the form of *data*. By applying a given ML *algorithm* $\mathcal{A}$, e.g. Random Forest (RF), to analyze a given *dataset* $\mathcal{D}$, it is possible to *train* a ML *model* $\mathcal{M}$ that is able to 'predict' previously unseen data. We provide a schematic of such workflow in Fig. 2.2. In the case of PWD, a ML model $\mathcal{M}$ can be deployed in a detector (e.g., in the hexagon in Fig. 2.1) to *infer* whether a given webpage is benign or phishing.

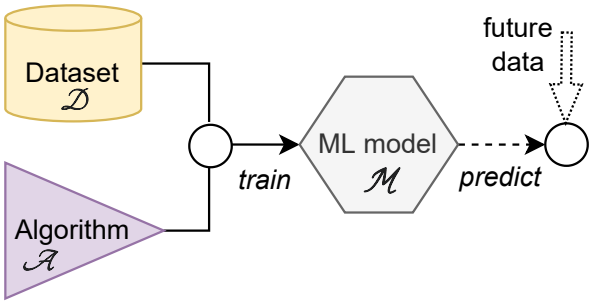


FIGURE 2.2: Machine Learning workflow. By training $\mathcal{A}$ on $\mathcal{D}$, a ML model $\mathcal{M}$ is developed. Such $\mathcal{M}$ can be used to predict future data.

The main advantage of ML models is their intrinsic ability of noticing weak patterns in the data that are overlooked by a human, and then leveraging such patterns to devise 'flexible' detectors that can counter even adaptive attackers. As a matter of fact, Tian et al. [247] show that a ML model based on RF is effective even against "squatting" phishing websites—while retaining a low-rate of false alarms (only 3%). Moreover, acquiring suitable data (i.e., recent and labelled) for ML-PWD is not difficult—compared to other cyber-detection problems for which ML has been proposed [62].

Such advantages have been successfully leveraged by many research efforts (e.g., [196, 241]). Existing ML-empowered PWD can leverage different types of *information* (i.e., *features*) to perform their detection. Such information can pertain either to a website's *URL* [255] or to its *representation*, e.g., by analyzing the actual image of a webpage as rendered by the browser [131], or by inspecting the HTML [144]. For example, Mohammad et al. [188] observed that phishing websites usually have long URLs; and often contain many 'external' links (pointing to, e.g., the legitimate 'branded' website, or the server for storing the phished data), which can be inferred from the underlying HTML. Although some works use only URL-related features (e.g., [85]) – which can also be integrated in phishing *email* filters (e.g., [128]) – more

recent proposals use combinations of features (e.g., [97, 252]); potentially, such features can be derived by querying third-party services (e.g., DNS servers [143]).

The cost-effectiveness of ML-PWD increased their adoption: even commercial browsers (e.g., Google Chrome [171]) integrate ML models in their phishing filters (which can be further enhanced via customized add-ons [242]); moreover, ML-PWD can also be deployed in corporate SIEM [136]. However, it is well-known that no security solution is foolproof: in our case, ML models can be thwarted by exploiting the so-called adversarial attacks [58].

2.1.3 Adversarial Attacks against ML

The increasing diffusion of ML led to question its security in adversarial environments, giving birth to "adversarial machine learning" research [80, 91]. Attacks against ML exploit *adversarial samples*, which leverage perturbations to the input data of a ML model that induce predictions favorable to the attacker. Even imperceptible perturbations can mislead proficient ML models: for instance, Su et al. [237] modify a single pixel of an image to fool an object detector; whereas Apruzzese et al. [57] evade botnet detectors by extending the network communications with few junk bytes.

An adversarial attack is typically described with a *threat model*, which explains the relationship of a given *attacker* with the *defender's system*. In particular, the attacker has a *goal* and, by leveraging their *knowledge* and *capabilities*, they will adopt a specific *strategy* [80]. Common terms associated with the attacker's knowledge are *white-box* and *black-box*: in the former, the attacker knows everything about the defender; whereas in the latter the attacker knows nothing [207, 275]. The capabilities describe how the attacker can interact with the target system, e.g., they: can influence only the *inference* or also the *training* stage of the ML model; can use the ML model as an "oracle" by inspecting the output to a given input; and can be subject to constraints on the creation of the adversarial perturbation (e.g., a limited amount of queries).

Despite thousands of papers focusing on this topic, a universal and pragmatic solution has not been found yet. Promising defenses are invalidated within the timespan of a few months (e.g. distillation was proposed in [209] and broken in [89]). Even "certified" defenses [149] can only work by assuming that the perturbation is bounded within some magnitude—which is not a constraint to which real attackers must abide (as pointed out by Carlini et al. [88]). From a pragmatic perspective, *any defense has a cost*: first, because it must be developed; second, because it can induce additional overhead. The latter is particularly relevant in cybersecurity, because it may decrease the performance of the ML model when no adversarial attack occurs. For instance, a well-known defense is *feature removal* [235], which entails developing ML models that do not analyze the features expected to be targeted by a perturbation. Doing this, however, leads to less information provided to the ML model, hence inducing performance degradation (e.g., [58]). Even when countermeasures have a small impact (e.g., [104]), this is not negligible in cyber-detection: attacks are a "needle in a haystack" [247], and even a 1% increase in false positives is detrimental [253]. Therefore, ML engineers will not devise any protection mechanism unless the corresponding threat is shown to be dangerous in reality [156].

The Problem. Unfortunately, research papers intrinsically impair the development of secure ML systems, because the aim is often to "outperform the state-of-the-art". In adversarial ML, this leads to papers that either showcase devastating attacks stemming from extremely powerful adversaries (i.e., white-box [237]); or viceversa,

i.e., show that even oblivious attackers can thwart ML systems [207]. However, real 'adaptive' attackers (i.e., those that ML methods should be protected against) do not conform to these two extremes. Indeed, having complete knowledge of the target system requires a huge resource investment (especially if such system is devoted to cybersecurity), which may be better spent elsewhere; conversely, it is unlikely that opponents will launch attacks while knowing nothing of the defender. Hence, to provide *valuable* research, efforts on adversarial ML should start focusing on the gray area within these two extremes—which implicitly are more likely to occur [56]. In the context of ML-PWD, our paper is a first step in this direction: as we will show, evasion attempts evaluated in literature (§2.6), despite being devastating, are costly to launch—even in black-box settings.

2.2 The Evasion-space of Adversarial Attacks against ML-PWD

We aim to spearhead valuable research in adversarial attacks against ML-PWD. To this purpose, we first elucidate the internal functionalities of a ML-PWD (§2.2.1). Then, we propose our original formalization of the *evasion-space* of adversarial perturbations (§2.2.2). Finally, we explain why our contribution validates *all* prior work (§2.2.3).

2.2.1 Analysis of a ML-PWD

Let us connect the previously introduced concepts (cf. §2.1.1 and §2.1.2) and provide an overview of a generic ML-PWD in Fig. 2.3.

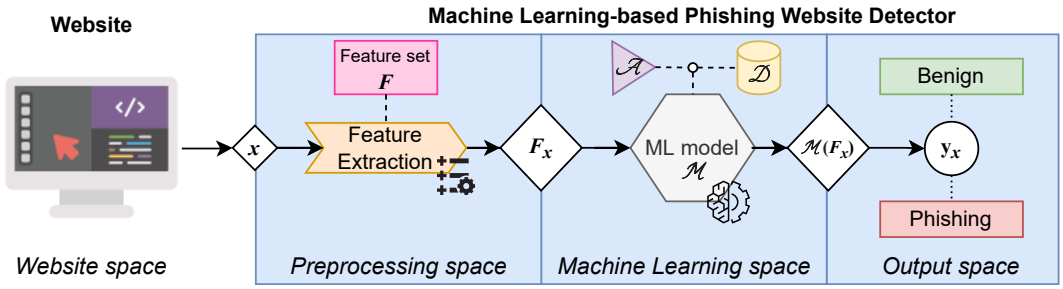


FIGURE 2.3: Architecture of a ML-PWD. A website, x, is preprocessed into F_x. A ML model $\mathcal{M}$ analyzes such feature representation and predicts its ground truth as $\mathcal{M}(F_x) = y_x$.

A sample (i.e., a website), x, 'enters' the ML-PWD and is subject to some preprocessing aimed at transforming any input into a format accepted by the ML model—according to a given feature set, F. (We assume that x is not blocklisted.) The result of such preprocessing is the feature representation of the website x, i.e. F_x, which can now be analyzed by the ML model $\mathcal{M}$. We consider a ML model focused on *binary classification*. Hence, training $\mathcal{M}$ requires: a dataset, $\mathcal{D}$, whose samples are labelled as *benign* or *phishing*; and any ML algorithm, $\mathcal{A}$, supporting classification tasks (e.g., RF).

The ML model $\mathcal{M}$ predicts the ground truth of F_x as y_x, i.e., $\mathcal{M}(F_x) = y_x$. Hence, we can summarize the workflow of our ML-PWD through the following Expression:

$$x \rightarrow F_x \rightarrow \mathcal{M}(F_x) = y_x. \tag{2.1}$$

If x is a phishing (benign) webpage and y_x is also phishing (benign), then we have a true positive (true negative); otherwise, we have an incorrect classification (either a false positive or a false negative). We assume that $\mathcal{M}$ has been properly trained, so that its deployment performance yields a high true positive rate (tpr) while maintaining a low false positive rate (fpr)—under the assumption that no adversarial attack occurs.

2.2.2 Evasion Attacks against ML-PWD

Adversarial attacks exploit a perturbation, ε, that induces a ML model $\mathcal{M}$ to provide an output favoring the attacker (cf. §2.1.3). In our case, $\mathcal{M}$ is a (binary) classifier that analyzes F_x, hence we can express an adversarial attack through the following Expression:

$$\text{find } \varepsilon \text{ s.t. } \mathcal{M}(F_x) = y_x^{\varepsilon} \neq y_x. \tag{2.2}$$

In other words, the objective is finding a perturbation ε that induces a ML model $\mathcal{M}$ (that is assumed to work well) to misclassify a given sample x (i.e., $y_x^{\varepsilon} \neq y_x$). Because our focus is on *evasion* attacks, such misclassification entails having a positive (i.e., phishing) classified as a negative (i.e., benign). It is implicitly assumed that such ε must: (i) preserve the *ground truth*[2] (i.e., y_x^{ε} should be the same as y_x); and (ii) preserve the *phishing logic* of a webpage [206]. Such ε, however, can lead to different effects on y_x^{ε} depending on 'where' it is applied during the workflow described by Exp. 2.1. We describe such occurrence by formalizing the *evasion-space* of an attacker.

Evasion-Space. Let us observe Fig. 2.3. We can see that the figure is divided into four 'spaces', each allowing the introduction of a perturbation ε that can affect the output of the ML-PWD. Of course, a perturbation in the last space, i.e., the *output-space*, cannot be considered as an 'adversarial ML attack', because it will have no relationship with the ML model $\mathcal{M}$. Hence, the evasion-space of an attacker that wants to induce a misclassification by $\mathcal{M}$ is confined to the first three spaces. Let us analyze each of these.

1. *Website-space Perturbations (WsP).* The entire detection workflow begins in the 'website-space', in which the website (i.e., x) is generated. Such space is accessible by any attacker, because they are in control of the generation process of their (phishing) website. As an example, the attacker can freely modify the URL or the representation of a website (subject to physical constraints[3]). Introducing a perturbation ε in this space (i.e., a WsP) yields an adversarial sample $\bar{x} = x + \varepsilon$, and the effects of such ε *can* affect all the operations performed by the ML-PWD (cf. Exp 2.1). We emphasize the word "can": this is because what happens *after* $\bar{x}$ enters the ML-PWD strictly depends on the implementation of such ML-PWD—which may, or may not, 'notice' the corresponding ε (e.g., $\mathcal{M}$ can analyze an F that is not influenced by ε).

2. *Preprocessing-space Perturbations (PsP).* After x is acquired by the ML-PWD, it is first transformed into F_x. *An attacker with write access* to the 'preprocessing-space' can introduce a PsP ε that affects the *process* that yields the feature representation of a website, leading to $\overline{F_x} = F_x + \varepsilon$. For instance, a website x with an URL of 40 characters can be turned into a $\overline{F_x}$ that has the *URL_length* feature=20. Intuitively, attackers able to introduce PsP are powerful, but are

[2] E.g., changing a URL from "go0gle.com" to "google.com" is not a valid ε.

[3] Which depend on the semantics of websites, e.g., URLs cannot be 1 character long.

still subject to constraints: before any F_x is sent to the ML model $\mathcal{M}$, such F_x is checked to ensure that it is not corrupted [56]. Indeed, $\overline{F_x}$ must not violate any inter-feature dependencies or physical constraints. With respect to WsP, PsP are guaranteed to be 'noticed' by the ML-PWD; however, they do not necessarily influence the predictions of $\mathcal{M}$: making a URL shorter may not be enough to fool the detection process.

3. *ML-space Perturbations (MsP).* After the preprocessing, the feature representation of a website F_x enters the Machine Learning-space in order to be analyzed by $\mathcal{M}$. *If an attacker has write access* to this space, they can introduce an MsP, i.e., a perturbation ε that affects F_x *immediately before* it reaches $\mathcal{M}$. An MsP is the 'strongest' type of perturbation because it affects the F_x after all integrity checks[4] have been performed—potentially leading to corrupted values, or which have no relationship to any real x. We hence denote MsP as $\bar{F}_x = F_x + \varepsilon$. As an example, a MsP can yield a $\bar{F}_x$ having an *URL_length*=0. As such, MsP are very likely to induce uncanny responses by $\mathcal{M}$ (but do not guarantee evasion).

Summary and Cost. From Exp. 2.2, we observe that any perturbation ε should ultimately affect the feature representation F_x of a given sample x. Hence, the crux is determining 'where' such perturbation is introduced—which can happen in three spaces. We formally define adversarial attacks by means of introducing a perturbation in each of these spaces (i.e., WsP, PsP and MsP) through the following Expression (which extends Exp. 2.1):

$$\text{find } \varepsilon \text{ s.t. } \begin{cases} \bar{x} = x + \varepsilon \;\Rightarrow\; x \to \bar{x} \to F_{\bar{x}} \to \mathcal{M}(F_{\bar{x}}) = y_x^{\varepsilon} \neq y_x & \text{WsP} \\ \overline{F_x} = F_x + \varepsilon \;\Rightarrow\; x \to \overline{F_x} \to \mathcal{M}(\overline{F_x}) = y_x^{\varepsilon} \neq y_x & \text{PsP} \\ \bar{F}_x = F_x + \varepsilon \;\Rightarrow\; x \to F_x \to \bar{F}_x \to \mathcal{M}(\bar{F}_x) = y_x^{\varepsilon} \neq y_x & \text{MsP} \end{cases} \tag{2.3}$$

We remark that the effects of WsP can match those of PsP—which can also match those of MsP. For instance, a MsP can yield a sample with an *URL_length* of 20 which – as long as it does not violate any inter-feature dependency – can represent a valid website (hence MsP=PsP)[5]; to obtain an equivalent WsP, the attacker would have to modify the actual URL and make it of exactly 20 characters (which is doable). Hence, in some cases, $F_{\bar{x}}$=$\overline{F_x}$=$\bar{F}_x$. As such, although some MsP cannot be crafted in the website-space, it is also unfair to consider all MsP (or PsP) as being not physically realizable. Finally, from a *cost* viewpoint, WsP$\ll$PsP$<$MsP, because realizing MsP requires the attacker to have more control[6] on the ML-PWD (i.e., they must obtain write-access to deeper segments of the ML-PWD).

2.2.3 Validation of Previous Work

An important contribution of our evasion-space is that it *validates all past research* that consider perturbations in the "feature-space" (i.e., PsP or MsP). Let us explain why.

[4]Indeed, a ML model $\mathcal{M}$ is agnostic to the generation process of a given input.

[5]Of course MsP=PsP if there is no 'integrity check'.

[6]Our formalization is orthogonal to the one by Šrndic and Laskov. [277]: while [277] focus on the attacker's *knowledge* ("what does the attacker know about the ML system?"), we focus on the *capabilities* (i.e., "where can the attacker introduce a perturbation affecting the ML system?"). Moreover, our PsP are semantically different than the "adversarial preprocessing" by Quiring et al. [217]: while [217] affect the preprocessing phase *from outside* the ML system, our PsP affect such phase *from the inside*.

Context. By using Pierazzi et al. [214] notation, our WsP can be seen as perturbations in the "problem-space"; whereas PsP and MsP are perturbations in the "feature-space". The main thesis of Pierazzi et al. [214] is that evaluations carried out in the feature space are unreliable due to the "inverse mapping problem": some changes in the feature representation of a sample (i.e., F_x) may not be physically realizable when manipulating the original sample (i.e., x)—therefore exposing the "weakness of previous evasion approaches."

Intuition. Our original formalization elucidates that the "weaknesses" of past work are not, in fact, weaknesses—therefore overturning some of the claims of Pierazzi et al. [214]. Our thesis is rooted in the following observation: the "inverse mapping problem" is irrelevant *if the attacker has write access to the ML-PWD*.

Explanation. Any attacker is able to craft WsP by manipulating their own phishing webpages (to some degree). In contrast, *reliably* realizing PsP and MsP can only be done by assuming an attacker that can manipulate the corresponding space (i.e., either the preprocessing- or the ML-space). Achieving this in practice presents a high barrier of entry—but *it is not impossible*. For instance, consider the case of an attacker who has compromised a given device integrating a client-side ML-PWD: such attacker can interfere with any of the ML-PWD operations—especially if it is open-source (e.g., [137]). Of course, realizing PsP or MsP if the ML-PWD is deployed in an organization-wide intrusion detection system is harder, but not unfeasible (as pointed out by [56]).

Takeaway: Our formalization validates all evasion attacks against ML-PWD previously evaluated through perturbations in any internal 'space' of the ML-PWD. This requires to *change the attacker's assumptions*, implicitly increasing the cost of the attack.

Consequences. Simply put, we *restore* the value (partially 'lost' after the publication of [214]) of the evaluations performed by prior work (§2.6). By assuming that the considered attacker can access a given space of the ML-PWD (either for PsP or MsP), then there is no risk of falling into the "inverse mapping problem"—because it is a constraint that such attacker is not subject to. Such different assumptions, however, implicitly raise the cost of the corresponding attack. For example. Corona et al. [97] craft perturbations in the ML-space: according to [214], the resulting perturbations are, hence, unreliable. However, by assuming that the attacker *can manipulate the ML-space*, then such adversarial examples (deemed unreliable by [214]) would become realistic (thanks to our contribution).

2.3 Proposed Realistic Threat Model

We use our evasion-space formalization to devise our proposed adversarial ML threat model—describing attractive strategies for real phishers. We first provide its definition (§2.3.1), and then support its realisticness via security analyses (§2.3.2). In Appendix 2.9 we show how to apply WsP on *real* phishing webpages. .

2.3.1 Formal Definition

We define our threat model according to the following four criteria (well-known in adversarial ML [80]).

Goal. The adversary wants to *evade* a ML-PWD that uses $\mathcal{M}$ as a detection method (i.e., the attacker wants to satisfy Exp. 2.2).

Knowledge. The adversary has *limited knowledge* of the target system, the ML-PWD. They know nothing about: the ML model $\mathcal{M}$, its training data $\mathcal{D}$, and its underlying ML algorithm $\mathcal{A}$ (except that it supports binary classification). However, the adversary knows a subset of the feature set F analyzed by $\mathcal{M}$. Let $K \subseteq F$ be such a subset. The adversary is also aware that the ML-PWD will likely detect phishing websites if no evasion attempt is made (otherwise, there would be no reason to do so). Finally, the adversary implicitly knows that no blocklist includes their phishing webpages (otherwise, the attacker would be *forced* to manipulate the URL).

Capability. The adversary has *no access* to the ML-PWD. They cannot use the ML-PWD as an "oracle" (i.e., inspect the output to a given input); and they are therefore confined to perturbations in the website-space (i.e., WsP).

Strategy. The adversary uses their knowledge of K to craft WsP that may result in successful evasion attacks *at inference time*.

We observe that our threat model is *general* because no specific set of features (F) or ML model $\mathcal{M}$ (and hence $\mathcal{D}$ and $\mathcal{A}$) is provided. Therefore, our threat model can cover any ML-PWD that resembles the one in Fig. 2.3. Potentially, it can even be a ML-PWD used by email filters if the corresponding $\mathcal{M}$ analyzes URL-related information (e.g., [110, 128]). Furthermore, our threat model can be *extended*. We will do so in our evaluation (§2.4), in which we compare the effects of attacks using WsP against those entailing PsP and MsP (by assuming the same knowledge, i.e., limited to K).

2.3.2 Security Analysis

Let us analyze our threat model and explain why it portrays a *realistic* attacker—especially if compared to typical 'white-/black-box' adversarial scenarios (cf. §2.1.3). We intend to justify that our threat model describes attacks that are *interesting* to investigate, and hence *valuable* for the security of ML-PWD.

Phishing in a nutshell. We start by focusing the attention on the intrinsic nature of phishing. Indeed, phishing attempts – and especially those involving phishing websites – are 'cheap' in nature [152]. Considering that real attackers operate with a cost-benefit mindset, it is unlikely that such attackers will invest extensive resources just to have their webpages evade a ML-PWD. Firstly, because such evasion will be temporary (as soon as the webpage is reported in a blocklist, any adversarial attack will be useless); secondly, because, even if a website evades a ML-PWD, the phishing attempt is not guaranteed to succeed (a user still has to input its sensitive data). Indeed, despite the exponential proliferation of phishing [28], most phishing attempts are prone to failure [200]—and the attackers are well aware of this fact. Of course, attackers can opt for more expensive spear-phishing campaigns [86] (which still have a success rate of barely 10% [134]), but in this case they will likely design entirely new phishing webpages—and not rely on cheap perturbations on pre-existing samples.

Limited Knowledge. Our attacker knows *something* (i.e., K) about the ML-PWD, but they are not omniscient—hence, our threat model can be considered as a gray-box scenario. Such 'box', however, is the entire ML-PWD, i.e., the blue rectangle in Fig. 2.3. Our scenario is *more interesting* to investigate than white-box scenarios. The reason is simple: ours is *more likely* to occur, because 'phishers' with complete knowledge of the entire ML-PWD are extremely unlikely. Furthermore, extensive adversarial ML literature [80] has ably demonstrated that white-box attacks can break most systems—including ML-PWD (e.g., [44, 124, 174, 236]).

Realistic Capabilities. Our 'standard' attacker has no access to the ML-PWD, which is a realistic assumption. For instance, the attacker can share a phishing website via social media, but without knowing which device (and, hence, ML-PWD) is being used by potential victims to open such website. Therefore, the attacker cannot reliably use $\mathcal{M}$ as an oracle. They could opt for querying a surrogate ML-PWD to reverse-engineer its functionalities and then leverage the transferability of adversarial attacks [105]. However, such 'black-box' scenario is both (i) unlikely to occur; and (ii) ultimately not interesting to consider for a research paper. *Unlikely*, because it would *defeat the purpose* of phishing attacks: reverse-engineering operations require a huge resource investment—which can be invalidated via a simple re-training of $\mathcal{M}$ (a common cybersecurity practice [61]). *Not interesting*, because such attacks *have been investigated before* [48, 221]. For instance, Liang et al. [171] clearly demonstrated that attackers with access to client-side detectors can successfully crack and evade the corresponding ML-PWD; doing this, however, required *more than 24 hours* of constant queries [171].

Takeaway: Phishing attempts have an intrinsic low rate of success. Attackers that aim to evade a ML-PWD will favor 'cheap' tactics—which can be represented by our proposed threat model.

Consideration. Attacking ML-PWD through (potentially unreliable) WsP is not the only way to 'realistically' evade ML-PWD. This is clearly evidenced by prior work—whose validity is restored thanks to our evasion-space formalization. However, our proposed 'cheap' attacks (through WsP) have never been investigated before in adversarial ML literature on PWD (§2.6). We hence set out to proactively assess the impact of feasible WsP on state-of-the-art ML-PWD; and comparing such impact to 'less realistic' (hence, less likely to occur) attacks performed through PsP and MsP. Therefore, our evaluation will also consider such worst-case scenarios. We stress, however, that our threat model shall not envision attackers who: (i) can observe or manipulate $\mathcal{D}$ (for poisoning attacks); (ii) can observe the output-space (for black-box attacks); (iii) have full knowledge of the ML-PWD (for white-box attacks).

2.4 Evaluation

As a constructive step forward, we assess the robustness of 18 ML-PWD against 12 evasion attacks—all based on our threat model, but performed in different evasion spaces. We have three goals:

- assess state-of-the-art ML-PWD against *feasible attacks*;
- compare perturbations introduced in *distinct evasion-spaces*;
- provide a *statistically validated benchmark* for future studies.

Achieving all such goals is challenging in research. Indeed, *crafting* perturbations in the three distinct spaces (i.e., WsP, PsP, MsP) requires: (i) datasets containing raw-data (for WsP), which are difficult to find; (ii) devising custom feature extractors (for developing the ML-PWD); as well as (iii) foreseeing the effects of WsP on such extractor (for PsP). Furthermore, to derive statistically sound conclusions, we must repeat our experiments multiple times [62].

We describe our experimental setup (§2.4.1), and then summarize our evaluation workflow (§2.4.2). More details are in Appendix 2.11.

2.4.1 Experimental Setup

We consider a total of 18 ML-PWD, which vary depending on the *source dataset* (2), the *ML algorithm* (3), and the *feature set* (3) used to develop the corresponding ML model. Such a wide array allows one to draw more generalizable conclusions.

Source Datasets

We rely on two datasets for ML-PWD: δ`phish` and `Zenodo` [97, 252]. Our choice is based on three reasons.

- Both datasets include *raw information* of each sample (specifically, its URL and its HTML). This is necessary because most of our attacks leverage WsP, for which we must modify the raw webpage, i.e., before its features are extracted.
- Both datasets have been *used by the state-of-the-art*. Prior research [97, 252] has demonstrated the utility of both datasets for ML-PWD, allowing for fair and significant comparisons.
- They enable experimental *reproducibility*. Indeed, collecting ad-hoc data through public feeds (e.g., AlexaTop/PhishTank) prevents fair future comparisons: phishing webpages are taken down quickly, and it is not possible to retrieve the full information of webpages 'blocklisted' years before.

We provide an overview of our datasets in Table 2.2, which shows the number of samples (benign and phish) and the performance (tpr and fpr) achieved by their creators (in the absence of evasion).

ML Algorithms

We consider ML-PWD based on shallow and deep learning algorithms [58] for binary classification. Our selection aims to provide a meaningful assessment of exemplary ML-PWD based on exemplary ML methods. In particular, we consider:

- Logistic Regression (LR). One of the simplest ML algorithms, we consider LR because it was (assumed to be) used by the ML-PWD embedded in Google Chrome [171].
- Random Forests (RF). An ensemble technique, RF often outperforms other contenders in phishing detection tasks [247].
- Convolutional neural Network (CN). We consider this well-known deep learning technique [163] due to its demonstrated proficiency also in ML-PWD (e.g., [258]).

Feature Sets

We consider ML-PWD that use three feature sets (F), all resembling the one described in our use-case (Appendix 2.9). Specifically, our ML-PWD analyze one of the following:

- URL-only (F^u), i.e., the first 35 features in Table 2.1.
- Representation-only (F^r), i.e., the last 22 features in Table 2.1.
- Combined (F^c), corresponding to all features in Table 2.1.

Rationale. Analyzing more information (i.e., larger feature sets, such as F^c) leads to superior detection performance—as shown, e.g., in [97]. However, in some cases this may not be possible: for instance, phishing *email* filters may make their decisions only by analyzing the URL (cf. §2.1.2). Nevertheless, modifying the URL is one of the easiest ways to trick a ML-PWD [201]: hence, a defender may develop an 'adversarially robust' detector that analyzes only the representation of a webpage. Such detector will have a lower performance (w.r.t. F^c) in non-adversarial scenarios, but will counter evasion attacks that manipulate the URL (cf. §2.1.3).

Observation. Our feature sets are not only popular in research (e.g., [130, 143, 188, 227]), but also used in *practice*. Indeed, several leading security companies yearly organize MLSEC, an ML evasion competition [22]. In 2021 and 2022, MLSEC also involved evading ML-PWD *which specifically analyzed the HTML* representation of a webpage—i.e., our F^r. We will also refer to MLSEC in our evaluation.

Considered Attacks

In our evaluation, we assess the robustness of each of the 18 ML-PWD against a total of 12 evasion attacks, which vary depending on the attacker's knowledge (i.e., K), capabilities (i.e., the evasion-space) and strategy (i.e., the features 'targeted'). In particular, we consider two macro-families of attacks:

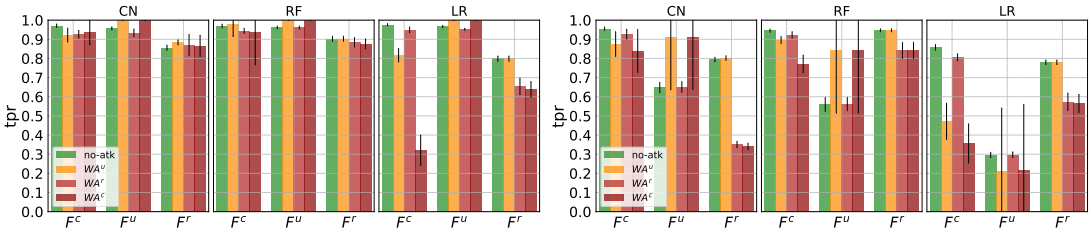
- **Cheap (Website) Attacks** (WA), corresponding exactly to our threat model and exhaustively described in our case-study (in Appendix 2.9). The adversary has no access to the ML-PWD, and can only apply WsP (which may not be effective).
- **Advanced Attacks**, where we relax some of the assumptions of our threat model to describe a more powerful attacker[7]. We consider three families: $\widehat{\text{WA}}$, wherein the attacker uses WsP, but knows a portion of the low-level implementation of the feature extractor; PA, wherein the attacker has write-access to (parts of) the *preprocessing-space*, and applies PsP; and MA, wherein the attacker has write-access to the *ML-space* and will apply MsP (a worst-case scenario).

Each of these four attack families (i.e., WA, $\widehat{\text{WA}}$, PA, MA) comes in three variants—depending on the features known (and targeted) by the attacker (i.e., u, r, c). For instance, WA^r is a WA in which the attacker tries to affect (through WsP) features related to the HTML representation of the webpage. Despite all our perturbations being ultimately 'blind' (the attacker will never be able to observe their effect), we can expect that MA will have a greater impact than WA on the ML-PWD. However such impact is compensated by the higher entry barrier for MA (see §2.2.2). More details, including a high-level estimate of the affordability of our attacks, are in Appendix 2.11.

2.4.2 Workflow and Statistical Validation

Each source dataset (`Zenodo` and `δphish`) represents a different setting—which we use to extract the corresponding training and inference partitions for our ML-PWD. Such ML-PWD are based on one among three ML algorithms, encompassing either shallow (*LR* and *RF*) or deep learning (*CN*) classifiers. Each of these classifiers

[7] These attacks are solely for research: their implicit higher cost w.r.t. WA may discourage real phishers from launching them (although they are not completely impossible).



(A) Impact of WA on the ML-PWD trained on Zenodo.

(B) Impact of WA on the ML-PWD trained on δPhish.

FIGURE 2.4: Effectiveness of the most likely attacks (WA). The three plots in each subfigure represent the algorithm used by a specific ML-PWD. Each plot has bars divided in three groups, each denoting a specific F used by the ML-PWD. The green bars show the tpr on the original samples, while the others show the tpr against a specific variant of WA.

presents three variants, depending on the analyzed features (F^u, F^r, or F^c), yielding a total of 9 'baseline' ML-PWD per source dataset. After ensuring that such 9 ML-PWD maximize their performance (high tpr and low fpr, at least for F^c), we assess their robustness against *all* the 12 proposed evasion attacks. Such attacks come in four families (WA, $\widehat{\text{WA}}$, PA, MA) depending on the knowledge and capabilities of the opponent, and each family presents three variants denoting the specific strategy, i.e., which features are 'targeted' by the attacker (either u, r, or c). We consider ML-PWD using F^c to be the 'true' baselines (likely highest performance in the absence of evasion attempts); whereas those using either F^u or F^r can be considered as 'robust' baselines (i.e., those using F^u will protect against attacks targeting F^r, and viceversa).[8] Such workflow is depicted in Fig. 2.6.

To provide results that are devoid of experimental bias and also to serve as a reliable benchmark for future researches, *we repeat all the abovementioned operations 50 times*. This means that each source dataset is randomly sampled 50 times, each resulting in a different training partition $\mathcal{D}$ and, hence, a different $\mathcal{M}$. Such $\mathcal{M}$ is, in turn, assessed on different data (i.e., different inference partitions), yielding different tpr and fpr, and is also subject to the 12 evasion attacks (all using different malicious samples as basis).

Such a large[9] evaluation allows one to perform *statistically validated comparisons* by leveraging well-known techniques [62]. We will do this to infer whether some attacks induce a performance degradation that is statistically significant. To the best of our knowledge, we are the first to use statistical tests to validate the impact of adversarial attacks against ML-PWD.

2.5 Results and Discussion

We present the results of our evaluation by focusing on our evasion attacks. Specifically, our results aim at answering two questions:

- (§2.5.1) how dangerous are the most likely attacks (i.e., WA)?
- (§2.5.2) what is the effectiveness of attacks carried out in different evasion spaces (i.e., $\widehat{\text{WA}}$, PA, MA)?

[8] Of course, the attacker expects the target ML-PWD to be using F^c.

[9] Overall, for our experiments we develop 900 $\mathcal{M}$ (given by: 2 source datasets * 50 random draws * 3 F * 3 $\mathcal{A}$), each assessed against 1200 adversarial examples.

We discuss our evaluation and potential for future work in §2.5.3. Our Artifact includes the full 'benchmark' results.

Preliminary assessment. Our results *in the absence* of adversarial attacks, reported in Table 2.3, show that the best ML-PWD on both datasets use *RF*. We appreciate that the 'true' baseline ML-PWD (using F^c) exhibit similar results as the state-of-the-art (cf. Table 2.2). In contrast, the 'robust' baselines (using either F^r or F^u) are slightly inferior[10]. For instance, on `Zenodo`, the *RF* using F^u has almost the same performance as F^c, but the one using F^r has 5% less tpr and 2% more fpr; whereas on δ`phish`, the *RF* using F^u has 50% less tpr (but similar fpr), while the one using F^r has 0.5% more fpr, but only 3% less tpr. Such degradation is the **cost** of using defenses based on *feature removal* on the considered ML-PWD. The expected benefit, however, is a superior resilience to evasion attempts.

2.5.1 Effectiveness of the most likely attacks (WA)

Let us focus the attention on the most likely attacks. We report in Figs. 2.4 the tpr achieved by all our ML-PWD against all our WA attacks (red bars), and compare it with the tpr (*no-atk*, shown in green bars) achieved by the same ML-PWD on the original set of samples used as basis for WA. Some intriguing phenomena occur.

True Baseline (F^c). We first consider ML-PWD using F^c (leftmost group of bars in each plot), as they are the 'true' baseline.

- On δ`phish` (Fig. 2.4b), all ML-PWD are affected by the 'strongest' cheap attack, i.e., WA^c. Specifically, the ML-PWD using *LR* is completely defeated (from 0.86 tpr down to 0.36); in contrast, those using *CN* or *RF* suffer a smaller, but still significant drop (from nearly 0.95 down to ∼0.8). Notably, the *CN* despite being worse than the *RF* in non-adversarial settings (cf. Table 3.6), appears to be slightly more robust.

- The situation is different on `Zenodo` (Fig. 2.4a). Here, while the *LR* is still defeated, the *CN* and *RF* appear not to be very affected by WA^c. However, considering that both *CN* and *RF* exhibit very high performance in non-adversarial settings (cf. Table 2.3), it is crucial to determine whether WA^c poses a real threat to such ML-PWD. To this purpose, we carry out a Welch t-test, which we can do thanks to our large amount of trials. We set our null hypothesis as "WA^c and *no-atk* are equal". The findings are valuable: against *RF*, the *p-value* is 0.221; whereas against *CN*, the *p-value* is 0.002. By using the common statistical significance threshold of 0.05, we can hence provide the following answer: the *RF is not affected* by WA^c, whereas the *CN is affected* by WA^c.

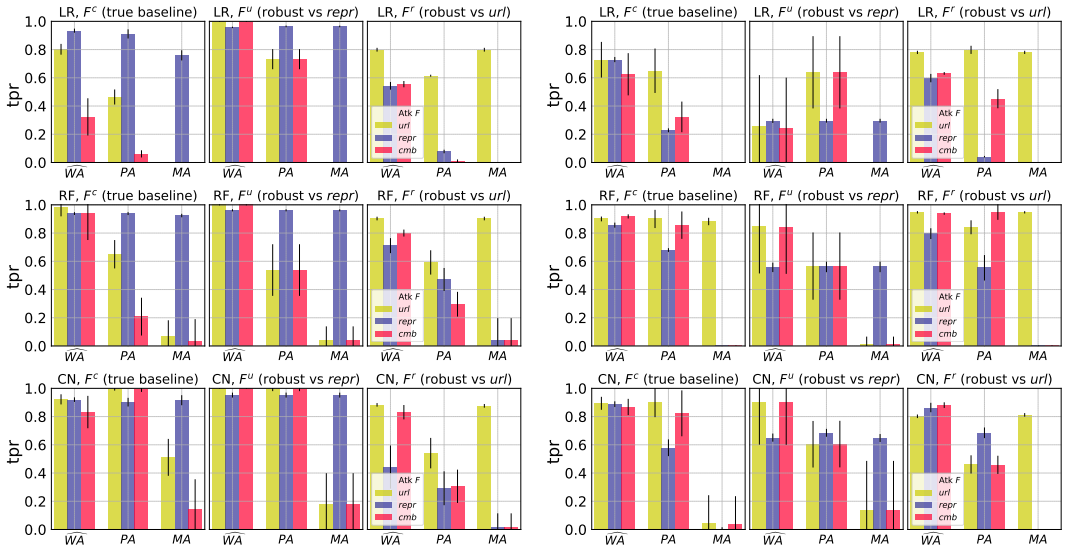
The latter finding is intriguing, because it suggests that *shallow learning methods can be more resilient* than deep learning ones for PWD—against our proposed attacks. Finally, we also observe that WA^r clearly defeat *LR* on both datasets, whereas the impact on *RF* and *CN* is significant on δ`Phish`, but small on `Zenodo`.

Robust Baselines (F^u, F^r). The robust baselines are, in general, reliable against WA. The ML-PWD using F^u counter WA^r (and viceversa), because the tpr is exactly the same as the original one. Notably, however, ML-PWD using F^r (similar to the

[10] Focusing on the ML-PWD using F^r (which are similar to the real ML-PWD in MLSEC [22]), we appreciate that *RF* achieves a remarkable 0.935 tpr and 0.01 fpr (averaged on both datasets), making such ML-PWD a valid baseline.

ML-PWD of[11] MLSEC [22]) are affected by WA^r: the *LR* is clearly defeated on both datasets, whereas *RF* suffers a 10% and 3% drop on δphish and Zenodo, respectively. Nevertheless, we observe a fascinating phenomenon: in some cases, the *tpr* under attack *is higher* than in *no-atk*; e.g., on δphish the *RF* analyzing F^u has its *tpr* to increase from 0.56 to ∼0.84 against both WA^u and WA^c. Such phenomenon occurs because the attacker (in any variant of WA) does not know 'what to do' to reliably evade the ML-PWD: the attacker guesses some WsP, which can have no impact, or even make the website closer to a 'malicious' one (from the viewpoint of $\mathcal{M}$).

Takeaway: The realistic attacks in the website-space (WA^c) can evade five (out of six) ML-PWD. Despite being small, the performance degradation is statistically significant: hence, due to their cheap cost, WA^c represent a threat to state-of-the-art ML-PWD.



(A) Zenodo. Each plot reports the *tpr* resulting from the 9 advanced attacks (i.e., $\widehat{\mathsf{WA}}$, PA, MA) across the 50 trials. Colors denote the targeted features (u, r, c).

(B) δphish. Each plot reports the *tpr* resulting from the 9 advanced attacks (i.e., $\widehat{\mathsf{WA}}$, PA, MA) across the 50 trials. Colors denote the targeted features (u, r, c).

FIGURE 2.5: Comparison of attacks carried out in different evasion-spaces. Each subfigure refers to a specific dataset, and presents 9 plots. Such plots are organized in three rows and three columns. Rows denote a specific ML algorithm (*LR*, *RF*, *CN*). Columns denote a specific feature set: the 'true' baseline (using F^c) is on the left; the others are the 'robust' baselines (using F^u or F^r).

2.5.2 Comparing the evasion-space ($\widehat{\mathsf{WA}}$, PA, MA)

We now focus on comparing the effectiveness of attacks that aim at influencing the same features (i.e., either u, r, c), but whose perturbations are introduced in different spaces (i.e., either WsP, PsP, or MsP). We visualize such results in Fig. 2.5.

The 'true' baselines (using F^c, i.e., the leftmost plots in Fig. 2.5) are defeated by MA. However, there are some notable exceptions: on Zenodo, the *RF* and *CN* are resilient to MA^r (this is because the HTML features have little importance for F^c).

[11] We also successfully attacked the competition-grade ML-PWD of [22] with WA^r, achieving similar results than the one shown in our custom-built ML-PWD. A demonstrative video (of 140s) can be found at the homepage of our website.

In contrast, on δphish, *RF* can withstand MA^u. The 'robust' baselines counter the corresponding MA, but unsurprisingly suffer against the others.

In general, PA tend to have a larger impact than $\widehat{\mathsf{WA}}$ against the 'true' baselines. However, this is not always true: we find enlightening that the *CN* on Zenodo is more robust to PA than to $\widehat{\mathsf{WA}}$. What is even more surprising is that such *CN* significantly outperforms the *RF* against PA, but *also* against MA. Such finding could inspire deployment of ML-PWD using deep learning on Zenodo—despite being inferior to *RF* in the no-atk (Table 2.3) and against WA^c (§2.5.1).

We note that $\widehat{\mathsf{WA}}^u$ perfectly match WA^u, which makes sense as they involve exactly the same WsP (cf. Appendix 2.11). We can also see some discrepancies between $\widehat{\mathsf{WA}}$ and PA: as a matter of fact, our anticipation of the preprocessing-space (i.e., the PsP of PA) did not exactly match what truly happened in the website-space . However, in some cases (e.g., the *RF* using F^c and F^r on δphish) we observe that the effectiveness of $\widehat{\mathsf{WA}}$ and PA tend to be similar. Such crucial finding demonstrates that perturbations applied directly to F_x (which we use for PA) can induce the same effects as those applied to x (which we use for $\widehat{\mathsf{WA}}$). In other words: if properly crafted, then even perturbations in the "feature-space" can resemble adversarial examples that are physically realizable [248].

Let us compare our attacks with those considered by δphish creators. Specifically, the attacks in [97] manipulate increasingly higher amounts of features (up to 10), and all ultimately evade target ML-PWD (which analyzes the HTML). Such finding is confirmed by our results on the ML-PWD analyzing F^r on δphish against MA^r, which all misclassify the adversarial samples. However, *if the perturbations are applied in different spaces* (i.e., PsP or WsP), *then the ML-PWD is significantly less affected.*

2.5.3 Discussion and Future Work

Our evaluation is a proof-of-concept, and we do not claim that *all* ML-PWD will respond in the same way as ours, and neither we claim novelty in the 'generic' method used to to evade PWD (attackers have been manipulating the HTML or URL for decades [80]). Indeed, our goal was to validate our primary contribution (whose focus is on machine learning) by performing a fair comparison of attacks (each having a different *cost*) in diverse evasion-spaces.

Warning on WA. A legitimate observation is that our cheap attacks, despite affecting most ML-PWD, have a small impact—even if statistically significant (§2.5.1). Such results, however, must not induce conclusions such as "these attacks are not interesting" or (worse) "these attacks can be overlooked in the security lifecycle". Indeed, *the main threat of* WA *is represented by the cheap cost*: thousands of phishing websites are created every day [28], and in such big numbers even a 1% difference can be the separation between a compromised and secure system [61]. Our goal is not to propose devastating attacks that bypass any ML-PWD; rather, we focus on those attacks that are more likely to occur in reality. As a matter of fact, WAs *can be automatized and implemented within seconds and few lines of code*; in contrast, the advanced attacks (including those of past work, e.g., [97, 171]) require to compromise or reverse-engineer the ML-PWD (§2.2.1). The *cost* of an attack should also account for the *effort* required for its implementation. Most related literature focuses on measuring 'queries' (e.g., [105]): our WA do not require any query. Nonetheless, we invite future work to explore metrics to estimate the cost of attacks in terms of human effort.

Extensions. The main purpose of our evaluation is to highlight how state-of-the-art ML-PWD respond to diverse evasion attacks. There are, however, millions

of ways to do the above. For instance, the attacks can target different features (and in different ways) than the ones considered in our evaluation (i.e., u, r, c); the ML-PWD can analyze different features, which can be generated via different preprocessing mechanisms (e.g., [154]). Additional defenses can also be considered (e.g., adversarial training [205, 250]). For instance, we did not consider ML-PWD that analyze the visual representation of a webpage (e.g., [44, 174]): such attacks would resemble those conducted in computer vision, which are well-known to be effective (e.g., [208, 249]). Nevertheless, our threat model is agnostic of the data-type, so we endorse future work to also consider ML-PWD analyzing images. Finally, our evasion-space formalization can be applied even to settings beyond phishing (e.g., malware), which may entail attackers more likely to use PsP or MsP.

2.6 Related Work

Countering phishing is a long-standing security problem, which can be considered as a subfield of cyberthreat detection—a research area that is being increasingly investigated also by adversarial ML literature [58]. We focus on the detection of phishing *websites*. Papers that consider phishing in social networks [81], darkweb [267], phone calls [129], or emails [110] are complementary to our work—although our findings can also apply to phishing email filters if they analyze the URLs included in the body text (e.g., [128]). Our focus is on attacks *against* ML-PWD. For instance, Tian et al. [247] evade PWD that use common blacklists, and their main proposal is to use ML as a detection engine to counter such "squatting" phishing websites. Hence, non-ML-PWD (e.g., [272]) are outside our scope.

Let us compare our paper with existing works on evasion attacks against ML-PWD. We provide an overview in Table 2.4, highlighting the main differences of our paper with the state-of-the-art. Only half of related papers craft their attacks in the problem-space—which requires modifying the raw webpage. Unfortunately, most publicly available datasets do not allow similar procedures. A viable alternative is composing ad-hoc dataset through public feeds as done, e.g., by [124] and [221] (the latter only for URL-based ML-PWD). All these papers, however, do not release the actual dataset, preventing reproducibility and hence introducing experimental bias. The authors of [236] share their dataset, but while the *malicious* websites are provided with complete information (i.e., URL and HTML), the *benign* websites are provided only with their URL—hence preventing complete reproducibility of attacks in the problem-space against ML-PWD inspecting the HTML. The latter is a well-known issue in related literature [206], which does not affect our paper because our entire evaluation is reproducible. Notably, Aleroud et al. [49] evaluate attacks both in the problem and feature-space, but on *different* datasets, preventing a fair comparison. Indeed, they evade one ML-PWD trained on `PhishStorm` (which only includes raw URLs) with attacks in the problem space; and another ML-PWD trained on `UCI` (which is provided as pre-computed features) through feature space attacks. Hence, it is not possible to compare these two settings. A similar issue affects also [48], which consider 4 datasets, each having a different F. Therefore, no prior work *compared the impact* of attacks carried out in distinct evasion-spaces—to the best of our knowledge. Not many papers consider adversarially robust ML-PWD, and only half consider both SL and DL algorithms—which our evaluation shows to respond differently against adversarial examples (cf. §2.5.2). It is concerning that few papers

overlook the importance of statistically significant comparisons. The most remarkable effort is [229] which only performs 10 trials (we do 50), which are not enough to compute precise statistical tests.

Nevertheless, most prior work assume stronger attackers than those envisioned in our threat model (cf. §2.3). Indeed, past threat models portray *black-box* attackers who can freely inspect the output-space and query the ML-PWD (e.g., [48, 171, 221]); or *white-box* attackers who perfectly know the target ML model $\mathcal{M}$, such as its configuration, its training data $\mathcal{D}$, or the feature importance (e.g., [44, 124, 174]). The only papers considering attackers that are closer to our threat model are [165, 201] and [44]. However, the ML-PWD considered in [44] is specific for *images*, which are tough to implement (cf. §2.5.3) and also implicitly resembles a ML system for computer vision—a task well-investigated in adversarial ML literature [80]. In contrast, the ML-PWD considered in [165] and [201] is similar to ours, but the adversarial samples are randomly created in the feature space, hence requiring an attacker with write-access to the internal ML-PWD workflow. Such an assumption is not unrealistic, but very unlikely in the context of phishing (cf. §2.3.2).

2.7 Conclusions

This study aims to provide a constructive step towards developing ML systems that are secure against adversarial attacks.

Specifically, we focus on the detection of phishing websites, which represent a widespread menace to information systems. Such context entails attackers that actively try to evade 'static' detection mechanisms via crafty, but ultimately simple tactics. Machine learning is a reliable tool to catch such phishers, but ML is also prone to evasion. However, realizing the evasion attempts considered by most past work requires a huge resource investment—which contradicts the very nature of phishing. To provide valuable research for ML security, the emphasis should be on attacks that are more likely to occur in the wild. We set this goal as our primary objective.

After dissecting the architecture of ML-PWD, we propose an original interpretation of attacks against ML systems by formalizing the EVASION-SPACE of adversarial perturbations. We then carry out a large evaluation of evasion attacks exploiting diverse 'spaces', focusing on those requiring less resources to be staged in reality.

TAKEAWAY: The findings of our paper are useful to both research and practice in the domain of adversarial ML.

- Our *evasion-space* formalization allows **researchers** to evaluate adversarial ML attacks without the risk of falling into the "unrealizable" perturbation trap (as long as the corresponding cost is factored in).
- Our *results* raise an alarm for **practitioners**: some ML-PWD can be evaded with simple tactics that do not rely on gradient computations, days of brute-forcing, or extensive intelligence gathering campaigns.

2.8 Supplementary Tables and Figures

We report in Table 2.1 the complete list of features of the ML-PWD considered in our paper. Table 2.2 shows some essential information on our datasets; Table 2.3 reports

the baseline performance of our ML-PWD (developed through the workflow shown in Fig. 2.6); and Table 2.4 shows the related works discussed in §2.6.

TABLE 2.1: Features F of the considered ML-PWD.

#	Feature Name	#	Feature Name	#	Feature Name
1	URL_length	20	URL_shrtWordPath	39	HTML_commPage
2	URL_hasIPaddr	21	URL_lngWordURL	40	HTML_commPageFoot
3	URL_redirect	22	URL_DNS	41	HTML_SFH
4	URL_short	23	URL_domAge	42	HTML_popUp
5	URL_subdomains	24	URL_abnormal	43	HTML_rightClick
6	URL_atSymbol	25	URL_ports	44	HTML_domCopyright
7	URL_fakeHTTPS	26	URL_SSL	45	HTML_nullLnkWeb
8	URL_dash	27	URL_statisticRe	46	HTML_nullLnkFooter
9	URL_dataURI	28	URL_pageRank	47	HTML_brokenLnk
10	URL_commonTerms	29	URL_regLen	48	HTML_loginForm
11	URL_numerical	30	URL_checkGI	49	HTML_hiddenDiv
12	URL_pathExtend	31	URL_avgWordPath	50	HTML_hiddenButton
13	URL_punyCode	32	URL_avgWordHost	51	HTML_hiddenInput
14	URL_sensitiveWrd	33	URL_avgWordURL	52	HTML_URLBrand
15	URL_TLDinPath	34	URL_lngWordPath	53	HTML_iframe
16	URL_TLDinSub	35	URL_lngWordHost	54	HTML_favicon
17	URL_totalWords	36	HTML_freqDom	55	HTML_statBar
18	URL_shrtWordURL	37	HTML_objectRatio	56	HTML_css
19	URL_shrtWordHost	38	HTML_metaScripts	57	HTML_anchors

All features in Table 2.1 are used by both the ML-PWD targeted in our pragmatic use-case (cf. §2.9), as well as by the 'true baselines' ML-PWD (i.e., those analyzing F^c) used in our evaluation (cf. §2.4.1); in contrast, the 'robust' ML-PWD (i.e., those analyzing either F^u or F^r) consider subsets of the features in Table 2.1 (see §2.4.1).

TABLE 2.2: Statistics and state-of-the-art of our datasets.

Dataset	#Benign	#Phish	*fpr*	*tpr*
δphish [97]	5511	1012	0.01	0.98
Zenodo [252]	2000	2000	0.08	0.99

We mention that the original Zenodo contains 100k phishing, and almost 4M benign webpages. To make our evaluation "humanly feasible," we randomly sample 4000 webpages from Zenodo, equally split between benign and phishing. In such a way, we can analyze the response of ML-PWD having diverse *balancing*: while Zenodo is perfectly balanced, δPhish has significantly more benign samples.

By comparing Table 2.3 with Table 2.2, we appreciate that our ML-PWD using F^c achieve comparable performance as prior work (even after our subsampling on Zenodo), confirming their relevance as baseline. Our repository includes the 4K pages we used for Zenodo.

TABLE 2.3: Performance in non-adversarial settings, reported as the average (and std. dev.) tpr and fpr over the 50 trials.

$\mathcal{A}$	F	Zenodo		δphish	
		tpr	fpr	tpr	fpr
CN	F^u	$0.96_{\pm 0.008}$	$0.021_{\pm 0.0077}$	$0.55_{\pm 0.030}$	$0.037_{\pm 0.0076}$
	F^r	$0.88_{\pm 0.018}$	$0.155_{\pm 0.0165}$	$0.81_{\pm 0.019}$	$0.008_{\pm 0.0020}$
	F^c	$0.97_{\pm 0.006}$	$0.018_{\pm 0.0088}$	$0.93_{\pm 0.013}$	$0.005_{\pm 0.0025}$
RF	F^u	$0.98_{\pm 0.004}$	$0.007_{\pm 0.0055}$	$0.45_{\pm 0.022}$	$0.003_{\pm 0.0014}$
	F^r	$0.93_{\pm 0.013}$	$0.025_{\pm 0.0118}$	$0.94_{\pm 0.016}$	$0.006_{\pm 0.0025}$
	F^c	$\mathbf{0.98}_{\pm \mathbf{0.006}}$	$\mathbf{0.007}_{\pm \mathbf{0.0046}}$	$\mathbf{0.97}_{\pm \mathbf{0.007}}$	$\mathbf{0.001}_{\pm \mathbf{0.0011}}$
LR	F^u	$0.95_{\pm 0.009}$	$0.037_{\pm 0.0100}$	$0.24_{\pm 0.017}$	$0.011_{\pm 0.0026}$
	F^r	$0.82_{\pm 0.017}$	$0.144_{\pm 0.0171}$	$0.74_{\pm 0.025}$	$0.018_{\pm 0.0036}$
	F^c	$0.96_{\pm 0.007}$	$0.025_{\pm 0.0077}$	$0.81_{\pm 0.020}$	$0.013_{\pm 0.0037}$

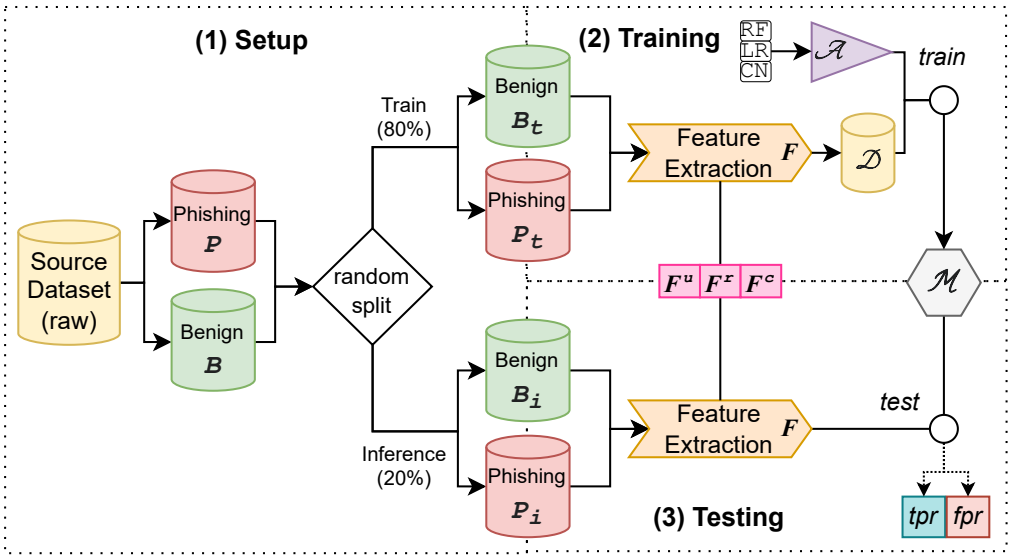


FIGURE 2.6: Experimental workflow. Each source dataset (containing benign, B, and phishing, P, samples) is randomly split into the training (B_t and P_t) and inference (B_i and P_i) partitions, used to train and test each ML-PWD. We use P_i as basis for our adversarial samples.

TABLE 2.4: Adversarial attacks against ML-PWD. For each paper, we report: the *evasion space* (for simplicity we consider problem and feature-space); which *features* (F) are analyzed by the ML-PWD; the ML *algorithms* used by the ML-PWD (SL or DL); if some *defense* is evaluated; how many *datasets* are used (and if they are reproducible); and if the experiments are repeated for *statistical validation*.

Paper (1st Author)	Year	Evasion space	ML-PWD types (F)	ML Algorithms	Defense	Datasets (reprod.)	Stat. Val.
Liang [171]	2016	Problem	F^c	SL	✗	1 (✗)	✗
Corona [97]	2017	Feature	F^r, F^c	SL	✓	1 (✓)	✗
Bahnsen [74]	2018	Problem	F^u	DL	✗	1 (✗)	✗
Shirazi [229]	2019	Feature	F^c	SL	✗	4 (✓)	✓*
Sabir [221]	2020	Problem	F^u	SL, DL	✓	1 (✗)	✗
Lee [165]	2020	Feature	F^c	SL	✓	1 (✓)	✗
Abdelnabi [44]	2020	Problem	F^r	DL	✓	1 (✓)	✗
Aleroud [49]	2020	Both	F^u	SL	✗	2 (✓)	✗
Song [236]	2021	Problem	F^c	SL	✓	1 (✓*)	✗
Bac [72]	2021	Feature	F^u	SL, DL	✗	1 (✗)	✗
Lin [174]	2021	Feature	F^c	DL	✓	1 (✓)	✗
O'Mara [201]	2021	Feature	F^r	SL	✗	1 (✓)	✗
Al-Qurashi [48]	2021	Feature	F^u, F^c	SL, DL	✗	4 (✓)	✗
Gressel [124]	2021	Feature	F^c	SL, DL	✓	1 (✗)	✗
Ours		Both	F^u, F^r, F^c	DL, SL	✓	2 (✓)	✓

2.9 Pragmatic Use-Case

Let us showcase *how* an attacker can physically realize WsP leading to adversarial samples. We intend to demonstrate that WsP "can be done", and hence represent a (likely) threat that must be considered in a proactive development lifecycle of ML-PWD.

Target System. We consider the ML-PWD proposed in [143], whose architecture aligns with the one in Fig. 2.3. The corresponding $\mathcal{M}$ is a *RF* classifier trained on a dataset created ad-hoc through public feeds. The complete feature set F analyzed by $\mathcal{M}$ is reported in Table 2.1, which includes features related to both the URL and the representation of the website (based on the HTML). The ML-PWD extracts such features by inspecting the raw webpage according to the thresholds proposed in [188] (and also used in [143]). We observe that such methodology (and, hence, F) is also adopted by very recent works (e.g., [130, 227]). We provide more details in the Artifact.

Attacker. The attacker expects the usage of a ML-PWD, but they are agnostic of anything about the ML model $\mathcal{M}$, i.e., they are oblivious of the ML algorithm (i.e., *RF*) and its training data. The attacker, however, follows the state-of-the-art and hence knows the most popular feature sets used by ML-PWD (e.g., [227]). In particular, the attacker correctly guesses that the ML-PWD analyzes features related to both the URL and the representation of the webpage, and specifically the URL length and the objects embedded in the HTML. Formally: K=(*URL_length, HTML_objectRatio*). The attacker, however, does not know the *exact* functionality of the feature extractor, the complete feature set F, and which features are more important for the final classification (the latter requires knowledge of $\mathcal{M}$). To provide a concrete example, we assume that the attacker owns the phishing[12] webpage shown in Fig. 2.7, whose URL is "https://www.63y3hfh-fj39f30-f30if0f-f392.weebly.com/".

Real Perturbations. To craft perturbations in the website-space (i.e., WsP) that affect $K \subset F$, the attacker can do the following.

[12] PhishTank reports such webpage to be a true and verified phishing (March 2022).

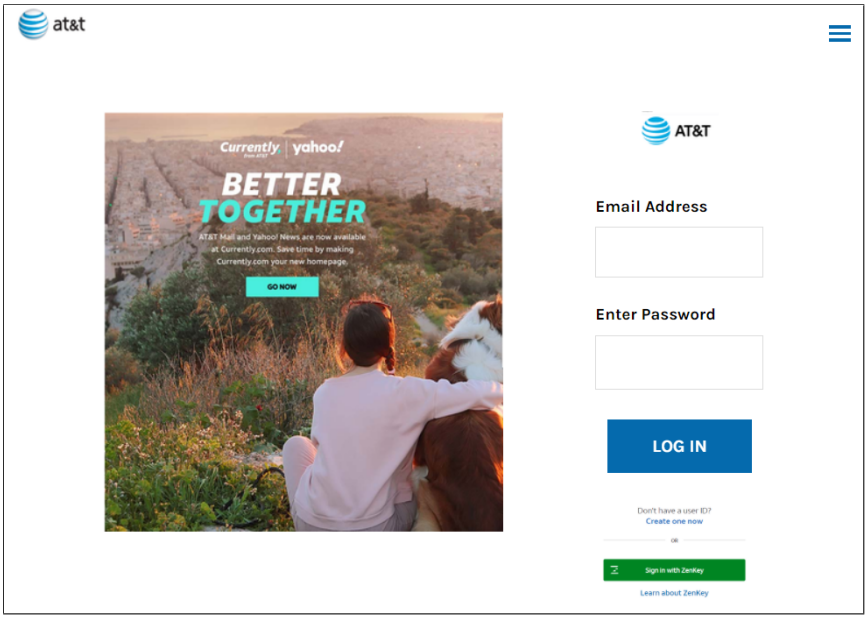


FIGURE 2.7: An exemplary (and true) Phishing website, whose URL is https://www.63y3hfh-fj39f30-f30if0f-f392.weebly.com/.

(1) *Modify the HTML.* The attacker knows that phishing websites have many links that point to external domains[13] with respect to internal resources (which would require to invest more into webhosting). Hence, the attacker can introduce (in the HTML) a high number of 'fake links' that point to non-existent internal resources, which will affect the ratio of internal-to-external objects (making it more even). Such fake links, however, are can be made invisible (by exploiting some CSS properties) to users, who will not notice any difference[14]. We provide a visual representation of such WsP in Fig. 2.8, showing a snippet of the HTML of the original phishing webpage (cf. Fig. 2.7); the red rectangles denote two exemplary 'perturbations', i.e., the introduction of (hidden) links pointing to an internal resource (which may not exist). Note that such WsP does not break the website's functionality, and can be cheaply introduced anywhere (and many times) in the source HTML. Similar WsP are feasible and will[15] influence the *HTML_objectRatio* (included in K).

(2) *Modify the URL.* The attacker knows that long URLs are suspicious. So the attacker can, e.g., use a URL-shortening service (e.g., bit.ly) to alter the length of the phishing URL. In our case, the original URL (of 52 characters) can be shrunk to "https://bit.ly/3MZHjt7" (of 14 characters), thereby resulting in a completely different URL. Such a WsP will affect many features analyzed by $\mathcal{M}$ (cf. Table 2.1). Such features are not included in K, and hence their modifications are beyond the attacker's knowledge. The shrunk URL can then be distributed by the attacker in the wild[16].

(3) *Both of the above.* The attacker can easily perturb both the URL and HTML to induce perturbations of higher impact.

[13] E.g., phishing associated with AT&T will have many links pointing to the real AT&T.

[14] N.b.: complete 'invisibility' is not a strict requirement. Some WsP can be 'spotted' by a detailed analysis, but users may not notice them while still being phished. E.g., a link can be deleted; or a WsP can wrap: *<a href='link'>* into *<a onclick="this.href='link'">*.

[15] In theory, similar WsP could be detected by analyzing whether a given link is valid or not. Doing so, however, would pose an extremely high overhead: it requires checking every single link for every webpage that is analyzed by the ML-PWD.

[16] The ML-PWD will be fooled if it is *stateless* and does not visit all the redirections of the shortening service. Nevertheless, there are many ways to reduce the URL_length.

Observation. None of these WsP are guaranteed to evade the ML-PWD. Indeed, a short URL is not necessarily benign, and having a non-suspicious ratio of internal-to-external objects is also not a strict requirement for being a benign webpage. The WsP could even be useless in the first place, e.g., the original URL could be already 'short'. Indeed, our attacker is not aware of what happens inside the ML-PWD. The problem, however, is that such uncertainty is shared by both the attacker (who cannot observe the ML-PWD) and the defender (who cannot exactly pinpoint what the attacker does). To reveal the uncanny effects of such WsP, we assess them in §2.5

FIGURE 2.8: A perturbation ε in the website-space (WsP). The original HTML (related to the website in Fig. 2.7) is modified by introducing *hidden link*(s). Such WsP will not be noticed by a user.

2.10 Threat Model: Considerations

Let us enhance our threat model with four considerations.

(1) The attacker can easily acquire a rough idea of the feature set F analyzed by the ML-PWD. For instance, the descriptions of many state-of-the-art solutions are openly accessible. However, it is unlikely that the attacker knows the *exact* feature set F: the actual implementation of a ML-PWD (including the feature extractor) can – or, rather, should! – differ from the publicly available information. This is why we consider an attacker that only knows $K \subseteq F$.

(2) We note that it is also possible that $K = \varnothing$. In this case, the attacker expects the ML-PWD to analyze some features that are *not* actually analyzed by $\mathcal{M}$ (for instance, the attacker can modify the URL, but nothing about the URL is analyzed by $\mathcal{M}$). This can happen, e.g., against an 'adversarially robust' ML-PWD that leverages the well-known *feature removal* strategy (cf. §2.1.3). As a result, WsP targeting such K will likely result in a negligible impact. Furthermore, it is also possible that some features in K simply *cannot* be influenced by an attacker operating in the website-space (e.g., features that depend on third-party sources, such as DNS logs).

(3) Since our attacker cannot access the ML-PWD, they cannot observe the output-space and, thus, cannot optimize their perturbations to find the best WsP that guarantees evasion; and cannot even verify whether their WsP evade the ML-PWD or not. The attacker is, however, not subject to strict boundaries on WsP (§2.2.2).

(4) Our threat model considers attacks at *inference*-time (i.e., after $\mathcal{M}$ has been deployed in the PWD). This is because the dataset used to devise ML-based security systems is typically well-protected [56]. Compromising such dataset would significantly raise the cost of the offensive campaign (as also highlighted in [172]). Therefore, phishers are unlikely to launch attacks at training-time.

The last two are significant: lack of access (and, hence, knowledge) on the training set prevents from achieving the no-box attacks of [166]; furthermore, the impossibility of witnessing the output of $\mathcal{M}$ prevents enacting typical black-box strategies (e.g., [194]).

2.11 Experiments: Considered Attacks

In our paper, we consider a total of 12 evasion attacks, divided in four families. One of these families is an *exact replica* of our 'standard' threat model. The remaining three families, however, are *extensions* of our threat model, which assume more 'advanced' adversaries who have superior knowledge and/or capabilities.

Two of our families involve WsP (WA and $\widehat{\text{WA}}$), but assume attackers with different knowledge; whereas the remaining two families involve either PsP or MsP (PA and MA). Each family has three variants depending on the features 'targeted' by the attacker, i.e., either those related to the URL, the HTML, or a combination of both (u, r, or c). For WsP, the underlying 'attacked' features are always the same for all variants, which are assumed to be known by the attacker: u is always the *URL_length*; for r is the *HTML_objectRatio*; and for c they are both of these. (Do note that our WsP will affect also features beyond the attacker's knowledge.)

- *Cheap Website Attacks* (WA) perfectly align with our threat model (ans resemble the use-cases in Appendix 2.9). The perturbations are created in the website-space (WsP), realizing either WA^u, WA^r, or WA^c. Specifically for r (and c), we consider two semantically equivalent WsP: "add fake link" for δ`Phish`, and "link wrapping" for `Zenodo`. Such WsP attempt to balance the object ratio: the former by adding (invisible) links to (fake) internal objects, whereas the latter by eluding the preprocessing mechanism—thereby having a link not being counted among the total links shown in a webpage.

- *Advanced Website Attacks* ($\widehat{\text{WA}}$), which envision a more knowledgeable attacker than WA. The attacker knows how the feature extractor within the ML-PWD operates (i.e., they know the specific thresholds used to compute some features). The attacker – who is still confined in the website-space – will hence craft more sophisticated WsP because they know how to generate an adversarial sample that is more likely to influence the ML-PWD. Thus, the attacker will modify either the URL, the HTML, or both (i.e., $\widehat{\text{WA}^u}$, $\widehat{\text{WA}^r}$, $\widehat{\text{WA}^c}$), but in more elaborate ways—e.g., by ensuring that the *HTML_objectRatio* exactly resembles the one of a 'benign' sample; or by making an URL to be 'long enough' to be considered short.

- *Preprocessing Attacks* (PA), which are an extension of our threat model, and assume an even stronger attacker that is able to access the preprocessing stage of the ML-PWD, and hence introduce PsP. Such an attacker is capable of direct feature manipulation—subject to integrity checks (i.e., the result must reflect a "physically realizable" webpage). Since the attacker does not know anything about the actual $\mathcal{M}$, the attacker must still guess their PsP. Such PsP will

target features based on either u, r, c (i.e., PA^u, PA^r, PA^c) by accounting for inter-dependencies between other features.

- *ML-space attacks* (MA), representing a worst-case scenario. The attacker can access the ML-space of the ML-PWD, and can hence freely manipulate the entire feature representation of their webpage through MsP. However, the attacker is still oblivious of $\mathcal{M}$, and must hence still guess their WsP. Thus, the MsP applied by the attacker completely 'flip' many features related to u, r, c (i.e., MA^u, MA^r, MA^c).

Motivation. We consider these 12 attacks for three reasons. First, to assess the effects of diverse *evasion attacks at increasing 'cost'*. For instance, the simplicity of WA makes them the most likely to occur; whereas MA can be disruptive, but are very expensive (from the attacker's viewpoint). Second, to study the response of ML-PWD to WsP targeting the same features (WA^r), but in different ways (one per dataset), leading to alterations of *different features beyond the attacker's knowledge*. Third. to highlight the *effects of potential 'pitfalls'* of related researches. Indeed, we observe that all three remaining families ($\widehat{\mathsf{WA}}$, PA, MA) envision attackers with similar knowledge which they use to target *similar features*. Such peculiarity allows comparing attacks carried out in different 'spaces.' A particular focus is on PA, for which we apply PsP by *anticipating* how a WsP can yield a physically realizable [248] PsP. Put differently, our evaluation shows what happens if the perturbations are applied without taking into account *all* preprocessing operations that transform a given x into the F_x analyzed by $\mathcal{M}$.

Implementation. We follow three steps: isolate, perturb, evade. We refer to the Artifact and source-code for the low-level details.

1. *Isolate*. Our threat model envisions evasion attacks that occur during inference, hence our adversarial samples are generated from those in P_i. Furthermore, we recall that the attacker expects the ML-PWD to be effective against 'regular' malicious samples (cf. §2.3.1). To meet such condition, we isolate 100 samples from P_i that are detected successfully by the best ML-PWD (typically using F^c). Such samples are then used as basis to craft all the adversarial samples (through WsP, PsP or MsP) of our evaluation—thereby ensuring that *all detectors are assessed against the exact same adversarial samples* (which is necessary for a fair comparison).

2. *Perturb*. We apply the perturbations as follows. For WA and $\widehat{\mathsf{WA}}$, we craft the corresponding WsP, apply them to each of the 100 samples from P_i, and then preprocess such samples by using the feature extractor. For PA and MA, we first preprocess the 100 samples with the feature extractor, and then apply the corresponding PsP or MsP. Overall, these operations result in 1200 adversarial samples (12*100).

3. *Evade*. The 1200 adversarial samples are then sent to all the 9 ML-PWD (for each dataset), and we measure the tpr again.

We expect the tpr on the adversarial samples (generated by any of our 12 considered attacks) to be lower than the tpr on the originals.

Effectiveness and Affordability. In terms of effectiveness, assuming the same targeted features, $\mathsf{WA} < \widehat{\mathsf{WA}} < \mathsf{PA} \ll \mathsf{MA}$ (§2.5.2). This is justified by the higher investment required by the attacker, who must either perform extensive intelligence gathering campaigns (to understand the exact feature extractor for $\widehat{\mathsf{WA}}$) or gain write-access to the ML-PWD (for PA and MA). Let us provide a high-level summary of the

requirements to implement all our attacks—all of which are *query-less* and rely on *blind* perturbations.

- WA: they require as little as a dozen lines of elementary code, and a very rough understanding of how ML-PWD operate (which can be done, e.g., by reading research papers).
- $\widehat{\text{WA}}$: they also require a few lines of code to implement. However, determining the exact thresholds requires a detailed intelligence gathering campaign (or many queries to reverse-engineer the ML-PWD, if it is client-side).
- PA: they require a compromise of the ML-PWD. For example, introducing a special 'backdoor' rule that "if a given URL is visited, then do not compute its length and return that the URL is *short*". Doing this is costly, but it is not unfeasible if the feature extractor is open-source (e.g., [73]).
- MA: they also require a compromise of the ML-PWD. In this case, the 'backdoor' is introduced *after* all features have been computed—and irrespective of their relationships. Hence. the cost is very high: the ML model is likely to be tailored for a specific environment, thereby increasing the difficulty of successfully introducing such backdoors in one of the deepest segments of the ML-PWD.

Hence, in terms of affordability: $\text{WA} \gg \widehat{\text{WA}} \gg \text{PA} > \text{MA}$ (i.e., the relationship is the reverse of the effectiveness). For this reason, in our evaluation we will put a greater emphasis on WA, because 'cheaper' attacks are more likely to occur *in the wild*: while WA can be associated with "horizontal phishing" (the majority), the others are tailored for "spear phishing" (the minority).

Chapter 3

Multi-SpacePhish: Extending the Evasion-space of Adversarial Attacks against Phishing Website Detectors using Machine Learning

In this Chapter, we will introduce the details of attack implementation in different spaces, and analyse the result of one proof-of-concept evaluation on competition-grade ML-PWD. Afterwards, we further propose two extensions of our threat model, i.e., *Deeper spaces* and *Mixed spaces*.

3.1 Attacks Implementation

We now focus on our considered attacks in a single space. We describe their technical implementation.

Let us discuss how we implement our perturbations, and provide some insight as to which features are influenced as a result of our attacks. We recall that each attack family presents three variants, depending on which features the attacker is 'consciously' trying to affect. Namely: u, r and c, i.e., features involving the URL, the representation (HTML) or a combination thereof. All attacks are created by manipulating (phishing) samples taken from P_i. In particular, during our first trial we isolate 100 samples from P_i that are correctly detected by the best ML-PWD: such samples are then used as basis for all their adversarial variants (to ensure consistency). For simplicity, we will denote any of such samples as p.

We start by describing MA which are the easiest to implement. Then, we describe WA and $\widehat{\text{WA}}$. Finally, we describe PA, which are the most complex to implement because they must consider several implications (e.g., inter-feature dependencies). (Our repository includes the exact implementation of MA and PA, and also all the pre-processed variant of the samples generated via WA and $\widehat{\text{WA}}$.)

ML-space attacks

The attacks (i.e., MA) are the easiest to implement. Indeed, we simply follow the same procedure as done by most prior works (e.g., [97, 165]) that directly manipulate the feature representation F_p of a sample p right before it is analyzed by the ML-PWD. We do this without taking into account any inter-dependency between features and/or any physical property that the actual webpage must preserve: this is compliant with our assumption that the attacker has access to the ML-space. Specifically, for each MA we apply the following MsP:

- MA^u: The attacker targets URL-related features. Hence, we manipulate F_p by setting features based on F^u equal to -1, which denotes a value that is more likely associated with a benign sample. In particular, we set to -1 the features in Table 2.1 with the following numbers: (1-17,19-21,27,30-35)
- MA^r: Same as above, but the targeted features are within F^r. Hence, we set to -1 the features in Table 2.1 with the following numbers: (36-40,42-52,54-57)
- MA^c: We set to -1 all features involved in MA^u and MA^r.

We remark that the attacker is not aware of the feature importance (because it would require knowledge of $\mathcal{M}$). Hence, although some manipulations will likely 'move' F_p towards a benign webpage, it is not guaranteed that $\mathcal{M}$ will actually classify such F_p as benign: if the manipulated features are not important, then even MsP may have no effect (and such phenomenon *does* happen in our evaluation, e.g., the ML-PWD using RF with F^c on `Zenodo` against MA^r).

Of course, we could set *all* features to -1 (e.g., all F^u and F^r). Doing this, however, would obviously result in a perfect misclassification (and hence not interesting to show). Moreover, it would not be sensible *even for the attacker*. Indeed, MA assume no knowledge of $\mathcal{M}$ and of $\mathcal{D}$, meaning that an attacker may suspect the existence of a honeypot [226]. For instance, $\mathcal{D}$ may contain some samples with all features set to -1 (i.e., benign) that are labelled as phishing—for the sole purpose of defeating similar attacks in the ML-space. Hence, it is realistic to assume that even an attacker capable of MA would not exaggerate with their perturbations.

Website attacks

We recall that we perform two families of attacks in the website-space: WA and $\widehat{\mathsf{WA}}$. The peculiarity of both of these attacks (both relying on WsP) is that the attacker does not have access to the ML-PWD. Hence, they are not able to manipulate F_p, and they are not even able to *observe* F_p.

- WA These attacks resemble the pragmatic example (§2.9).
  - WA^u: We set the URL to a random string starting with "www.bit.ly/", followed by 7 randomly chosen characters (which what this popular URL shortener does).
  - WA^r: For δ`Phish`, we change the HTML by adding 50 invisible internal links (i.e., having the same root domain of the website);[1] for `Zenodo`, we wrap all links within an "onclick", i.e., we change: *<a href='link'>* into *<a onclick="this.href='link'">*.[2]
  - WA^c: We do both of the above for each dataset.
- $\widehat{\mathsf{WA}}$: These attacks envision an attacker that knows how the feature extractor within the ML-PWD operates (see §2.4.1). Such knowledge can be acquired, e.g., if the attacker has (or is) an insider that provided them with such intelligence. However, the attacker is still confined in the website-space, and hence

[1] The exact string we inject is: "* can not see*", which is the second string shown in our pragmatic example (§2.9).

[2] This WsP, if applied to textual link, would remove the underline of such a link, therefore being visible to a user; however, it is possible to make it invisible by editing the CSS properties. Our feature extractor is agnostic of such properties, so we do not do this: the results would be equivalent.

can only apply WsP (to generate $\overline{p}$). For a meaningful comparison, we assume an attacker who is aware of how the features targeted in WA are "extracted" within the ML-PWD. Hence, we craft each $\widehat{\mathsf{WA}}$ as follows:

- $\widehat{\mathsf{WA}^u}$: The attacker, having knowledge of the extractor, knows that by using an URL shortener they will affect all features related to the URL (i.e., F^u); furthermore, they know the threshold (53) that makes an URL to be considered as 'benign'. Such length is well above that of an URL generated via any shortening service. As such, these attacks are an exact replica as $\widehat{\mathsf{WA}^u}$ (the only difference is that the attacker of $\widehat{\mathsf{WA}^u}$ is more confident than the one in WA^u).
- $\widehat{\mathsf{WA}^r}$: The attacker manipulates the HTML in the same was as in WA^c. However, the attacker also knows the threshold (0.15) of internal-to-external links that yields a benign value of the *HTML_objectRatio* feature. Hence, the WsP manipulate the HTML of each p by introducing as many links (or wrappings) as necessary to meet such threshold.
- $\widehat{\mathsf{WA}^c}$: The attacker does both of the above.

We stress that the attacker cannot observe $F_{\overline{p}}$. Indeed, doing this would require the attacker to completely replicate the feature extractor, which is costly, and may not even be possible (some third-party services may require subscriptions to be used). As such, the attacker is aware of how to craft WsP that are more likely noticed by the ML-PWD, but evasion is not guaranteed.

Preprocessing attacks

These attacks are the hardest to realize *from a research perspective* and *in a fair way*.

Challenges. The underlying principle of PsP (the backbone of PA) is affecting the preprocessing space of the ML-PWD. Technically, since we are the developers of our own feature-extractor (i.e., the component of the ML-PWD devoted to data preprocessing), we could simply directly manipulate our own extractor, i.e., by introducing a 'backdoor'. However, doing this would prevent a fair generalization of our results: for instance, it is possible to develop another feature extractor, having the same functionality but whose operations are executed in a different order. Hence, to ensure a more fair evaluation, we apply the perturbations *at the end* of the preprocessing phase, but we do so by anticipating how a perturbation in the website-space (a WsP) could affect the preprocessing-space, thereby turning a WsP into a "physically realizable" PsP. To this purpose, we *assume the viewpoint of an attacker*. For instance, we ask ourselves: "if an attacker wants to affect URL features by using an URL shortener, how would the feature extractor react?".

Scenario. In PA the attacker *knows* and *can interfere* (through PsP) with the feature extraction process of the targeted ML-PWD. However, the attacker is *not* aware of what happens next: the ML-space and the output-space are both inaccessible by the attacker (from both a *read* and *write* perspective). Hence, once the PsP has been applied and $\overline{F_p}$ is generated, the attacker cannot influence $\overline{F_p}$ any longer. For each PA we do the following:

- PA^u: we anticipate an attack that targets URL features, and specifically *URL_length*, by using an URL shortener. Hence, we can foresee that operations (in the website-space) can lead to alterations of *all* the features involved with the URL (i.e., F^u). For instance, doing this would make weird characters (if present) to disappear from the URL. However, doing this would induce

to alterations also to F^r. For instance, some objects originally considered to be 'internal' would become 'external'. Hence, we implement PA^u by setting the following features (from Table 2.1) to -1: (1-3,5,6,8,10-16,22,23,25,26,28-30), whereas the following features are set to +1: (4,27,36-38,41,44,48,52,54,56).

- PA^r: we anticipate an attack that targets features related to the representation of a website—in our case the HTML, and specifically the *HTML_objectRatio* feature. We foresee that an attacker can interfere with such feature in many ways, for instance by removing links, adding new ones, or changing those already contained in the webpage. All such changes will affect many features, such as the *HTML_freqDom*: because populating the HTML with (fake) internal links would change the 'frequent domains' included in the HTML. Such changes can also affect the links in the footer of the webpage (*HTML_nullLnkFooter*); or the anchors (*HTML_anchors*); but also others. We implement PA^r by setting the following features (from Table 2.1) to -1: (36–38,41,51,54,56,57); whereas we set (39,40) to 1 and 46 to 0.

- PA^c: they are a combination of the two above. We expect the attacker to use a URL shortener, and also infterfer with the HTML_objectRatio. However, we cannot simply set the features to the same values as PA^r and PA^u, because one of the two will prevail. In our case, shortening the URL will be 'stronger', because the URL will change (to that of the URL shortener) and hence the internal objects will become 'external'. Hence, we implement PA^c by setting the following features (from Table 2.1) to -1: (1-3,5,6,8,10-16,22,23,25,26,28-30), whereas the following features are set to +1: (4,27,36-38,41,44,48,52,54,56).

We remark that our PsP may not yield an $\overline{F_p}$ that is a perfect match with a $F_{\bar{p}}$ generated via WsP (i.e., those of $\widehat{\mathsf{WA}}$). Indeed, some inconsistencies may be present—likely due to 'inaccurate' anticipations from our (i.e., the attacker's) side. Such inconsistencies are sensible. An attacker with access to the preprocessing-space can theoretically *replicate* the entire feature extractor, and use it to exactly pinpoint how to generate PsP that are an exact match with WsP (i.e., $\overline{F_p}$=$F_{\bar{p}}$). However, doing this would be *very expensive*. Furthermore, it would *defeat the purpose* of using PsP: the attacker does not want that $\overline{F_p}$=$F_{\bar{p}}$, rather, they want a PsP that is 'stronger'; otherwise, why use PsP in the first place?

3.2 Proof-of-concept: attacks against a competition-grade ML-PWD

To further prove the impact of our 'cheap' attacks (i.e., WA), we tested them on a real ML-PWD that is used in a well-known Machine Learning Security Evasion Competition (MLSEC [22]). Such competition is held yearly, and is organized by leading tech-companies that provide cybersecurity services reliant on ML methods. The 2022 edition of MLSEC envisions a challenge in which participants are asked to *evade* ML-PWD. We took this opportunity to assess whether our attacks had any impact against such 'competition-grade' ML-PWD. Short story: they do. A demonstrative video can be found at the homepage of our website (which also includes the source-code).

3.2.1 Challenge

Participants of the phishing evasion challenge are given 10 'phishing' webpages, which are provided in their raw HTML form. The purpose of the challenge is to manipulate such webpages so that (i) they render exactly as the originals, and (ii) they evade a ML-PWD. Specifically, the organizers provide 8 different ML-PWD, which the participants can use as a black-box: by sending an input (i.e., the HTML of a phishing webpage), they are given an output (i.e., the probability that such webpage is malicious—according to the specific ML-PWD). Such ML-PWDs only analyze the HTML of the webpage (which must render exactly as the original). Put simply: the objective of the challenge is to tweak the HTML of the 10 webpages with imperceptible modifications that decrease the confidence of the 8 ML-PWD.

3.2.2 Method

Of course, the setting described above perfectly describes the black-box scenarios envisioned in adversarial ML papers: query the detector, and use the response as a guide to craft a more evasive phishing webpage. Our primary attacks (WA), however, are query-less. Because we are aware that the target ML-PWD analyzes the HTML (recall that this is an assumption of our threat model), we then craft our 'adversarial' phishing webpages by using exactly the same WA^r used in our paper for δPhish: we add 50 invisible internal links. We apply these WsP to all the 10 webpages *provided by the organizers of the challenge*, and then test whether they had any impact to the *real* ML-PWD involved in the challenge.

3.2.3 Results

By taking into account *all* webpages against *all* ML-PWD, our attacks induced a drop of 3.4% in the confidence of the ML-PWD, indicating that our WsP had some effect. However, while some ML-PWD were not very affected, others incurred a significant drop. Specifically, we focus our attention on the first and third ML-PWD provided by the organizers of MLSEC. The results of our proof-of-concept experiments are shown in Figs. 3.1. These graphs show phishing probability (y-axis) given as output by the corresponding ML-PWD for each of the 10 webpages of the challenge (x-axis). We report two bars: the blue bar are the results of the original webpages, whereas the red bars are the results after applying our WsP.

3.2.4 Analysis

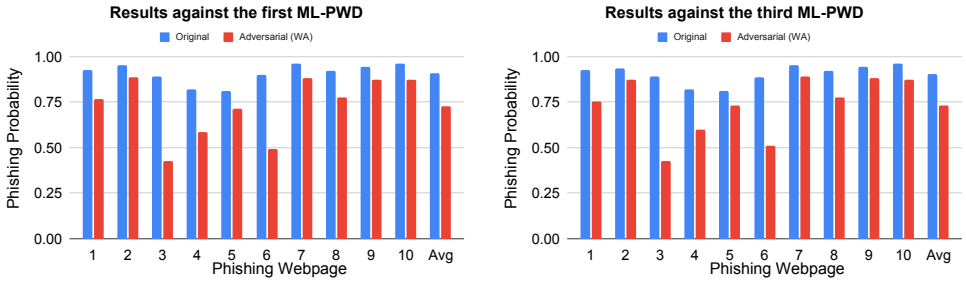
These two detectors were significantly less certain after our WsP, with an average confidence drop of 17.5%. We observe that in most cases, the confidences were still above 0.5 (i.e., the webpages would still be classified as 'phishing'). A more detailed look, however, reveals that **these detectors were completely fooled by some webpages** (i.e., their confidence dropped to below 0.5). We report:

- Page #3: from 0.90 down to 0.43 for the 1st and 3rd detectors.
- Page #6: from 0.90 down to 0.49 for the 1st detector.

We also attempted the same WA^r by changing the number of fake links, and also by considering a different string[3]. When applied to, e.g., webpage #3, adding 280

[3]We also considered the 'wrapping' WsP for Zenodo: the effects were negligible—probably because these ML-PWD factored such links into their 'count' (i.e., the attacker made a wrong guess). See Appendix 3.6.2

links dropped the confidence to below 0.2; whereas adding a slightly different string (the first one shown in our pragmatic example in Appendix B) 280 times, the confidence dropped to 0.2 for the first and third detector, and to 0.49 for the seventh detector. The seventh detector was also fooled by adding such alternative string 50 times to webpage #4, causing a confidence of 0.46 (down from 0.68). The source-code is available in our repository, and the experiments are entirely reproducible. Interestingly, these results align with those shown in our primary evaluation: our *query-less* WA attacks cannot bypass any ML-PWD, but in some cases *they can induce a miss-classification*.



(A) Impact of WA^r on the *first* ML-PWD.

(B) Impact of WA^r on the *third* ML-PWD.

FIGURE 3.1: Effectiveness of the most likely attacks (WA^r on $\delta\texttt{Phish}$) against the ML-PWD provided by the organizers of MLSEC [22].

3.3 Threat Model Extensions

Our threat model (§2.3) can be *extended* by relaxing some of its assumptions. Indeed, in its current formulation, our threat model envisions an attacker that is "weak" (and, hence, very likely to appear in reality). However, some adversaries may be willing to invest more resources to ensure that their attacks come to fruition (i.e., increasing the chances that their phishing webpages are misclassified by the ML-PWD, and hence displayed to the end-user). Abundant prior work in the adversarial ML domain considers attacks having different levels of *knowledge* (i.e., the so-called "black-box" and "white-box" [55]). However, given that our original formalization focuses on the attacker's *capabilities* (§2.2), we identify two types of extensions that portray a stronger attacker. Namely:

- *Deeper spaces.* An attacker who manages to obtain write-access to the ML-PWD (or part of its elements) can tamper with its internal functionalities, thereby realizing either PsP or MsP.
- *Mixed spaces.* If the attacker can obtain some control on either the Preprocessing- or Machine Learning-space, then – alongside being able to apply PsP or MsP – they are also able to apply WsP. Indeed, the attacker will *always* be able to manipulate the phishing webpage, since it is (by definition) under their complete control. Hence, an attacker who can inject PsP can also inject a WsP; furthermore, an attacker who can inject a MsP can also inject a PsP (since they can overlap), and can, of course, also inject a WsP.

We will empirically assess attacks entailing perturbations in different spaces (§3.4).

3.4 Additional Experiments: Same-space and Mixed-Space

We *expand* the evaluation carried out in Chapter 2 with additional experiments. Our goal is twofold:

- assessing other types of perturbations (either WsP, PsP or MsP) *in the same space*;
- consider a "stronger" attacker that applies *multiple* perturbations also in *different* spaces (cf. §3.3).

We first describe (§3.4.1) and empirically evaluate (§3.4.2) the attacks entailing perturbations in the "same-space". Then, we describe (§3.4.3) and evaluate (§3.4.4) the "multi-space" attacks.

3.4.1 Same-space Attacks: Description

In this section, we elaborate on new attacks in the same evasion space involving our WsP, PsP, and MsP. Building upon the attacks considered in the main evaluation (§2.11), we introduce additional perturbations. The motivation behind this extension is to present a more comprehensive range of use cases—all of which are likely to happen, since they are well within the attacker's capabilities (who will never have complete knowledge of the target PWD). Therefore, we explore novel perturbations of the HTML (§3.4.1) and URL (§3.4.1), as well as introduce new variations of MsP, PsP, and WsP. Altogether, the details of the new specific attacks are provided in Table 3.1.

HTML

As we know (§2.1), the HTML reflects the visual appearance of a webpage—therefore, changes to the HTML can lead to differences in the way the webpage is presented to its users.[4] Some of them may be noticed by users (e.g., alterations of the background), while others may not change the appearance at all (e.g., the hidden links considered in our pragmatic use-case §2.9). Here, we consider a wide-array of HTML-related perturbations, and scrutinize which are more likely to evade the detection of PWD. Practically, we propose a total of 37 new HTML-related perturbations—of which, 24 are WsP (i.e., new WA^r), which can be divided into the three following categories:

1) *iWsP* (invisible WsP), which denote perturbations that are inserted into the webpages but remain invisible to users. This means that the webpage appears unchanged before and after the perturbation insertion.

2) *eWsP* (elusive WsP), which introduce slight changes to the appearance of the webpage. While these changes may require some effort to be noticed by users, they are still discernible upon careful observation.

3) *rWsP* (recognizable WsP), which result in changes that are clearly visible by users. These modifications have a more pronounced impact on the webpage's appearance, making them readily noticeable.

[4]We recall that our threat model does not assume that the perturbations are "imperceptible" to humans. This is because, in a real scenario, phishing is effective because humans are distracted. Hence, even if the webpage changes, the phishing attack can still be successful.

The remaining 13 HTML-related perturbations are PsP and MsP (i.e., new PA^r and MA^r). Both of which require write-access to the ML-PWD. PA^r can bypass some of the checks of ML-PWD. Moreover, in MA^r, attackers may solely focus on evading ML-PWD: as a result, some MA^r might violate the fundamental rules of HTML.

URL

Domain and path are two essential components of URL, and most of our URL features in Table 2.1 are extracted from them. In this section, we implemented 6 types of perturbations that specifically target the URL. These perturbations, referred to as WA^u, the specific details are provided below.

- *replChar*, we replaced the characters in the domain with visually similar characters.
- *sepWrd*, we randomly inserted space within the domain to separate the individual word.
- *delChar*, we deleted one character from the domain.
- *swpChar*, we randomly swapped two adjacent characters in domain.
- *addChar*, we randomly inserted an additional character into the domain.
- *atkPth*, we also conducted operations of swap, delete, or insert randomly within the path of the URL.

We do not consider URL-related perturbations that affect other spaces (i.e., PsP or MsP).

3.4.2 Same-space Attacks: Evaluation

We now assess the impact of the abovementioned perturbations. For HTML perturbations (§3.4.2), we consider the effects both on the ML-PWD we developed by using the δ`Phish` and `Zenodo` datasets, as well as by those provided by MLSEC (we carried out these experiments in December 2022, when the MLSEC API was still open for research purposes). For the URL perturbations (§3.4.2) we consider only the ML-PWD trained on δ`Phish` and `Zenodo` because those provided by MLSEC do not consider the URL in their analyses.

Impact of HTML perturbations

We begin by considering δ`Phish`, `Zenodo`, and then focus on MLSEC.

δPhish and Zenodo. In Figs. 3.2, we present the *tpr* achieved by ML-PWD trained on δ`Phish` and `Zenodo`. We evaluate the performance of these ML-PWD against iWA^r, eWA^r and rWA^r (represented by yellow and red bars)[5]. To provide a comparison, we also include the *tpr* achieved by the same ML-PWD on the original set of samples, depicted by the leftmost green bar labeled as “no-atk". These results aim to address two key questions:

- Will different WsP have different impacts on ML-PWD and how?

[5]Our figures only present the most effective WsP, i.e., iWA^r denotes *addHidP*, eWA^r stands for *addImgBot*, and rWA^r represent *modFntClr*.

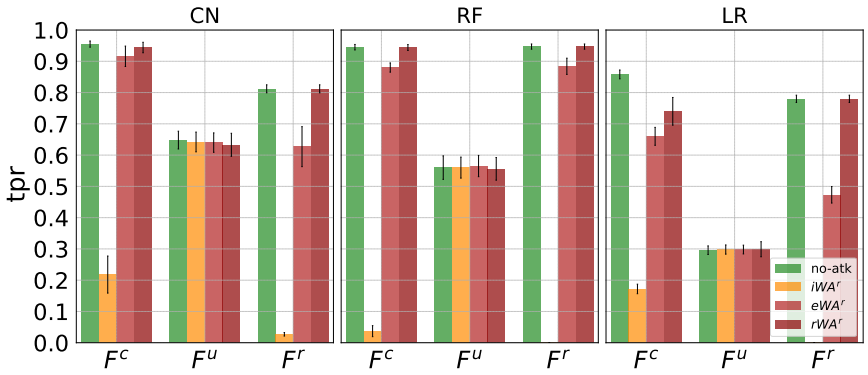
TABLE 3.1: New Attacks for HTML

Category	Perturbation	Description
iWA^r	*addInLnk*	insert internal links
	replOnc	$\langle a\ href = 'link'\rangle$ to $\langle a\ onclick = "this.href = 'link'"\rangle$
	delHidIt	delete hidden items from HTML
	addHidP	add hidden large page
	replJS	replace $\langle a\ href = '\#'\rangle$ with $\langle a\ href = 'javascript : void(0)'\rangle$
	replRet	replace '\n' with whitespace
	htEsc	escape the whole body content, and write "document.write(unescape(' '))" to HTML
	htEncd	encode HTML with base64
	replPass	replace $\langle input\ type = 'password'\rangle$ with $\langle input\ type = 'text'\rangle$
	replOnfoc	replace $\langle input\ type = 'password'/'email'\rangle$ with $\langle input\ onfocus = "this.type = 'password'/'email'"\rangle$
	addSusLnk	add suspicious links $\langle a\rangle$, e.g., $\langle a\ href = '\#skip'\rangle$
eWA^r	*addImgBot*	insert 20 small local images to the webpage bottom
	modFntTyp	modify the font type italic
	addTps	randomly insert few typos into HTML text
	modCpy	modify copyright
	addIcn	add local icon
	delSusLnk	delete suspicious links
	delSusFrm	delete suspicious form (i.e., with empty or external 'action' links)
	modTtl	randomly modify the title
	delCpy	delete copyright information from HTML
rWA^r	*modBgimg*	change the background image
	modBgClr	randomly change the background color
	modFntClr	randomly modify the font color
	modFntSiz	modify the body font size to 0
PA^r	*delTxt*	delete all text from HTML
	delFrm	remove forms
	delSpn	remove all span
	delTtl	remove title
	addLngTxt	add long visible text to HTML
	delFtr	remove footer
	replSusFtrLnk	replace suspicious links of footer with internal links
MA^r	*brTg*	break the tag $\langle html\rangle$
	delHt	remove the whole html
	delHd	delete the whole $\langle head\rangle$ except style
	delBdy	delete the whole $\langle body\rangle$
	brTgs	break tags
	hmg	replace characters with homographic letters

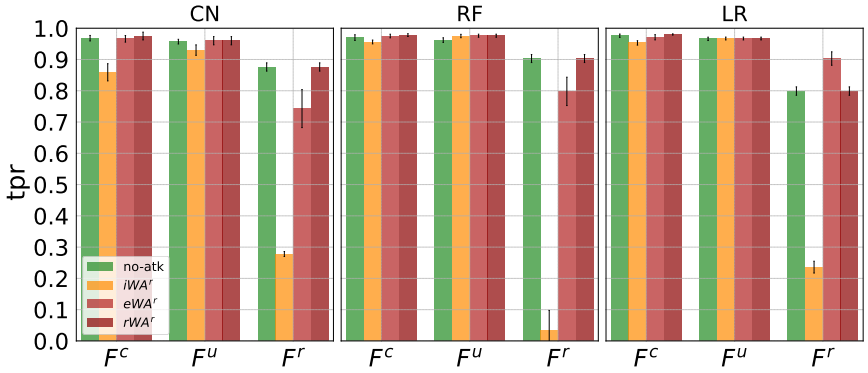
- What kind of WA is more likely to evade the ML-PWD trained on δPhish and Zenodo?

As shown in Fig. 3.2a, the iWA^r perturbation emerges as the most impactful attack, leading to a significant reduction (reduced by 0.68–0.95) in the tpr of F^r- and F^c-based ML-PWD trained on δPhish. Specifically, the tpr of RF-PWD trained on F^c drops from 0.945 to 0.037, and the tpr of RF-PWD trained on F^r decreases from 0.947 to 0. In comparison, the influence of eWA^r and rWA^{r}is relatively smaller. However, eWA^r still causes a notable drop in the tpr of F^r-based LR-PWD, reducing it from 0.78 to 0.47. On the other hand, rWsP has minimal impact on PWD (only F^c-based LR-PWD's tpr decreased by 0.12). A similar trend is observed in Fig. 3.2b for the influence on Zenodo, where iWA^r remains the most effective attack. Additionally, eWA^r affects ML-PWD to a greater extend (except for F^r-based LR-PWD) compared to rWA^r. These findings demonstrate that iWA^r poses the greatest challenge to ML-PWD of δPhish and Zenodo, significantly reducing their detection performance. eWA^r also has a notable impact, while rWA^r has a relatively minor effect on most ML-PWD (except for the ML-PWD using LR to analyze F^r).

Figs. 3.3 represents the impact of new PA^r and MA^r on ML-PWD trained on δPhish and Zenodo. In this context, PA^r refers to *delFrm* (i.e., remove forms from the webpage), while MA^r denotes applying perturbation *hmg* to HTML (i.e., inserting typos to the HTML, both tags and text). Comparing with the tpr of 'no-atk', it is evident that both PA^r and MA^r have negative impact on the tpr of ML-PWD trained on δPhish and *Zenodo*. Specifally, PA^r reduced the tpr of all F^c- and F^r-based ML-PWD on



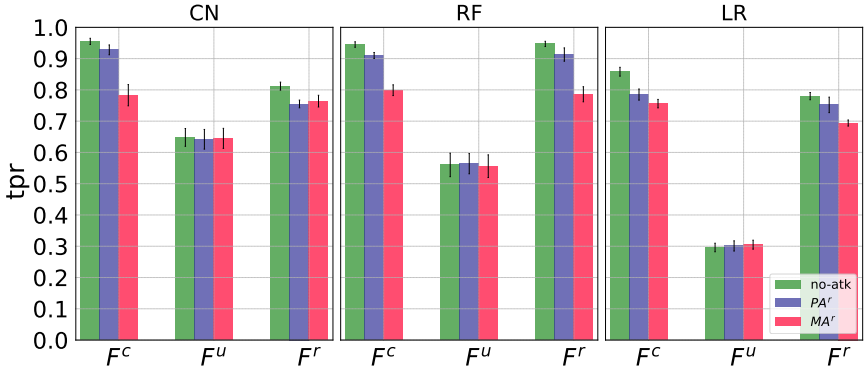
(A) Impact of new WA^r (i.e., iWA^r, eWA^r and rWA^r) on ML-PWD trained on δ`phish`



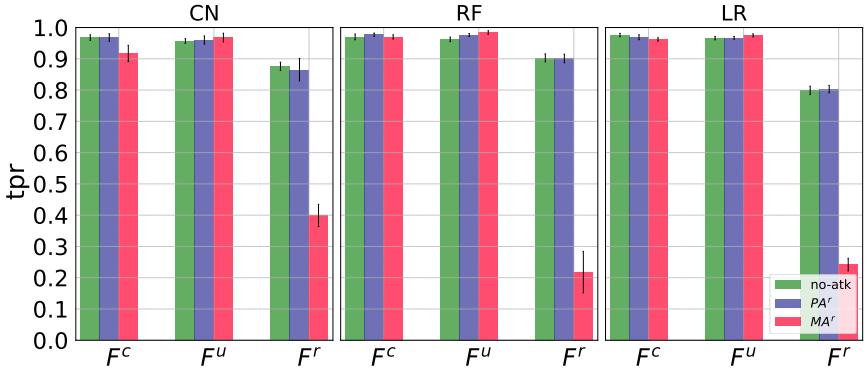
(B) Impact of new WA^r (i.e., iWA^r, eWA^r and rWA^r) on ML-PWD trained on `Zenodo`

FIGURE 3.2: Effectiveness of the most likely new attacks WA^r. The three plots in each subfigure represent the algorithm used by a specific ML-PWD. Each plot has bars divided in three groups, each bar denotes a specific F used by the ML-PWD. The green bars show the tpr on the original samples, while the others show the tpr against a specific variant of WA.

δPhish, with small decreases ranging from 0.01 to 0.08. On the other hand, MA^r had a more pronounced effect compared to PA^r, successfully reducing the tpr of F^r-based ML-PWD by 0.1–0.17. Nevertheless, WA^r is still the most effective attack compared with them.



(A) Impact of new PA^r and MA^r on ML-PWD trained on δphish



(B) Impact of new PA^r and MA^r on ML-PWD trained on Zenodo

FIGURE 3.3: Effectiveness of new attacks PA^r and MA^r. The three plots in each subfigure represent the algorithm used by a specific ML-PWD. Each plot has bars divided in three groups, each denoting a specific F used by the ML-PWD. The green bars show the tpr on the original samples, the blue bars represent tpr against PA^r and the red bars in the rightmost show the tpr against MA^r.

MLSEC. We have summarized the impact of the new HTML attacks on MLSEC in Table 3.2. These attacks are the same HTML attacks used in δPhish and Zenodo. Our findings reveal several interesting phenomena in the evaluation:

- Among the attacks evaluated, iWA^r emerges as the most potent attack, significantly degrading the performance of PWD of MLSEC. The confidence of models $m0$ and $m2$ drop from nearly 0.9 to 0.02, indicating a stark decrease in their ability to accurately detect malicious webpages. However, it is worth noting that other attacks also have a substantial impact on degrading the detection capability of PWD. For instance, PA^r reduce the confidence of $m2$ from 0.9 to 0.61, while MA^r results in a decrease of 0.76 in the confidence of $m6$.

- Comparing to eWA^r and rWA^r, iWA^r has a greater influence on $m0$–$m3$, leading to a decrease in their confidence by 0.35–0.89. However, for PWD $m4$–$m7$, iWA^r does not decrease their confidence but slightly increase them by 0.01. On the other hand, rWA^r reduces their confidence by 0.1 (from nearly 0.8 to 0.7), while eWA^r results in a confidence reduction of 0.2 for PWD $m4$ and $m6$. This phenomenon can be considered reasonable since PWD employed in MLSEC are

black-box models which may consist of multiple types of PWD. It implies that the impact of perturbations may vary depending on the specific model characteristics and vulnerabilities. Hence, it is important to note that the goal of this study is not to propose a generalized perturbation set that works for all PWD, but rather to investigate the impact and effectiveness of cheap perturbations on PWD in practice.

- It is observed that rWA^r has a more widespread impact as it influences all seven PWD on MLSEC, resulting in a reduction of confidence by 0.1 across the board.
- Both PA^r and MA^r are effective attacks that successfully evade the detection of PWD in MLSEC. In particular, MA^r proves to be a potent attack, as it evades five (out of eight) PWD, causing their confidence score to drop below 0.5. Additionally, the confidence scores of seven PWD decrease to approximately 0.65 from initial values of around 0.85. These findings highlight the impact of PA^r and MA^r on the performance of PWD on MLSEC.

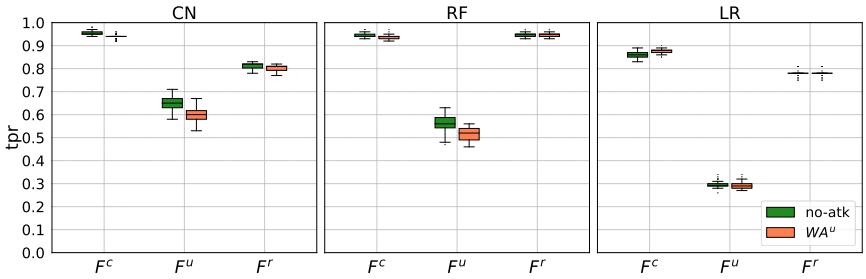
TABLE 3.2: New attack's impact on MLSEC (HTML perturbations)

$\mathcal{A}$	no-atk	iWA^r	eWA^r	rWA^r	PA^r	MA^r
$m0$	0.91$_{\pm 0.052}$	0.02$_{\pm 0.011}$	0.65$_{\pm 0.185}$	0.81$_{\pm 0.116}$	0.91$_{\pm 0.052}$	0.90$_{\pm 0.062}$
$m1$	0.87$_{\pm 0.071}$	0.52$_{\pm 0.161}$	0.87$_{\pm 0.085}$	0.78$_{\pm 0.100}$	0.67$_{\pm 0.262}$	0.31$_{\pm 0.051}$
$m2$	0.90$_{\pm 0.051}$	0.02$_{\pm 0.011}$	0.65$_{\pm 0.185}$	0.85$_{\pm 0.087}$	0.61$_{\pm 0.390}$	0.88$_{\pm 0.096}$
$m3$	0.88$_{\pm 0.070}$	0.51$_{\pm 0.172}$	0.87$_{\pm 0.079}$	0.81$_{\pm 0.091}$	0.66$_{\pm 0.271}$	0.26$_{\pm 0.080}$
$m4$	0.82$_{\pm 0.106}$	0.83$_{\pm 0.123}$	0.64$_{\pm 0.199}$	0.73$_{\pm 0.112}$	0.57$_{\pm 0.372}$	0.80$_{\pm 0.121}$
$m5$	0.81$_{\pm 0.120}$	0.82$_{\pm 0.136}$	0.85$_{\pm 0.107}$	0.70$_{\pm 0.103}$	0.64$_{\pm 0.280}$	0.39$_{\pm 0.166}$
$m6$	0.83$_{\pm 0.108}$	0.84$_{\pm 0.116}$	0.64$_{\pm 0.198}$	0.73$_{\pm 0.111}$	0.56$_{\pm 0.373}$	0.07$_{\pm 0.076}$
$m7$	0.82$_{\pm 0.121}$	0.83$_{\pm 0.127}$	0.85$_{\pm 0.106}$	0.70$_{\pm 0.097}$	0.64$_{\pm 0.279}$	0.36$_{\pm 0.129}$

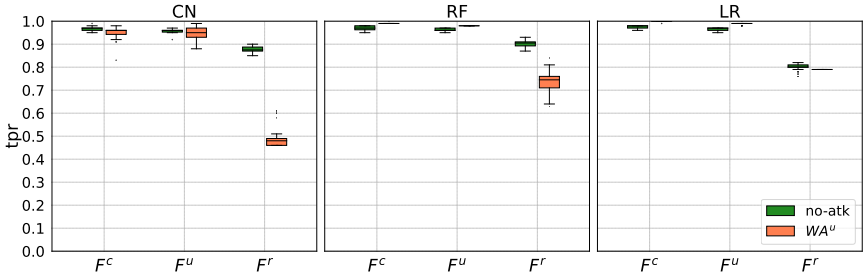
Takeaway: Applying iWsP does not change the webpage's appearance but it proves to be highly effective in evading ML-PWD. In contrast, the application of rWsP results in obvious changes to the webpage's appearance but it has a relatively minor impact on the performance ML-PWD. MA^r had a more pronounced effect compared to PA^r. Nevertheless, WA^r is still the most effective attack compared with them.

Impact of URL perturbations

The impact of WA^u is illustrated in Figs. 3.4. Fig. 3.4a reveal the changes when performing *atkPth* on ML-PWDtrained on δPhish. Green boxes represent the tpr of 'no-atk' (i.e., baseline), while the orange boxes indicate the impact of WA^u. Comparing the medians of each box plot, the median line of orange boxes is lower than Green boxes for F^u-based ML-PWD, indicating that WA^u can degrade ML-PWD's tpr. In contrast, this type of WA^u does not decrease tpr of F^u-based CN-PWD trained on Zenodo (as shown in Table 3.24 in Appendix 3.6.3. However, it is significantly reduces the performance of F^r-based ML-PWD. This is because some HTML features require extracting information from both URL and HTML (e.g., HTML_URLBrand: which checks (in the HTML) if the webpage title includes the brand name that appeared in the URL). Therefore, either URL perturbations or HTML perturbations can possibly affect the F^r-based ML-PWD. Furthermore, as shown in Fig. 3.4b, another WA^u *sepWrd*, also clearly decreases the tpr of F^r-based ML-PWD. Simply put, WA^u will affect ML-PWD's performance.



(A) Impact of new WA^u on ML-PWD trained on δphish



(B) Impact of new WA^u on ML-PWD trained on Zenodo

FIGURE 3.4: Effectiveness of new attacks WA^u. The three plots in each subfigure represent the algorithm used by a specific ML-PWD. Each plot has box divided into three groups, each denoting a specific F used by the ML-PWD. The green box shows the tpr on the original samples, while the orange box show the tpr against WA^u.

3.4.3 Multi-space Attacks: Description

Insofar, we have always considered perturbations applied in a single space. However, as mentioned in §3.3, an attacker who can apply PsP or MsP (which require write-access to the ML-PWD) can also apply WsP (which only requires access to the phishing webpage—which the attacker owns). These "mixed-space" attacks are worth considering because they are trivial to implement for an attacker—assuming that such an attacker can already apply PsP andor MsP (we recall that, from a cost viewpoint, WsP<< PsP<MsP). Therefore, we introduce 66 types of 'mixed-space' attacks (the complete details are in Appendix 3.6.3). These attacks span across all the defined evasion spaces (§2.2): Website space, Preprocessing space, and Machine Learning space. In particular, we consider "accessible" attacks (which combine WA and PA), as well as stronger ones (which entail MA and PA). We also consider "double" attacks, entailing multiple perturbations *in the same space* (e.g., WsP+WsP). We expect that mixed-space attacks, which exploit vulnerabilities and weaknesses present in different stages of the detection process, lead to more evasive samples (at least w.r.t. the corresponding single-space attacks).

3.4.4 Multi-space Attacks: Evaluation

We evaluate the evasion capabilities of our new mixed-space attacks on the ML-PWD trained on the δphish, Zenodo, as well as those provided by MLSEC.[6] We begin by considering attacks entailing two perturbations *in the same space*, i.e., PsP+PsP (§3.4.4) and WsP+WsP (§3.4.4); then, we consider attacks entailing two perturbations *in different spaces*, i.e., PsP+WsP (§3.4.4 and PsP+MsP (§3.4.4).

[6]Since MLSEC only analyzes the HTML, we do not consider mixed-space attacks entailing perturbations of the URL.

Double-PsP

- δ**Phish** and **Zenodo**. Table 3.12 and Table 3.26 demonstrate the impact of 8 kinds of PA^r+PA^r on ML-PWD trained on δPhish and Zenodo. Even though not all of them significantly impact the PWD of δPhish. While not all combinations significantly affect the PWD, there are notable influences observed. For instance, when the combination attack occurs(specifically, the perturbation delSpn_delTtl), the tpr of LR-PWD based on F^c and F^r drops by 0.1 and 0.16, respectively. additionally, the tpr of F^r-based LR-PWD down from 0.8 to 0.58, and CN-PWD's drops from 0.86 to 0.64 after being subjected to PA^r+PA^r. In contrast, F^u-based PWD is not affected, and most of F^c-based PWD remain unchanged. That is because our PA^r+PA^r combinations specifically target HTML, and F^u is the core component when crafting the F^c-based ML-PWD.

- **MLSEC**. In the case of MLSEC's PWD, Table 3.18 indicates that all cheap PA^r+PA^r combination attacks proposed can decrease the performance of PWD, resulting in the confidence score dropped by 0.01–0.32.

Double-WsP

- δ**Phish** and **Zenodo**. As shown in Table 3.14, the combination attack WA^r+WA^r did not reduce the tpr of ML-PWD trained on δPhish. In fact, in some cases, the tpr increased to 1.0, such as the tpr of F^r-based CN-PWD increased from 0.79 to 1. Similarly, '*replOnfoc_replRet*' did not affect the ML-PWD of Zenodo, as shown in Table 3.29). However, it is importance to note that under the influence of '*htEsc_replRet*', the tpr of F^r-based LR-PWD reduced to 0.55 from 0.8. Moreover, '*htEncd_replRet*' reduced tpr of F^r-based CN-, LR- and RF-PWD to 0. These findings suggest that while some combinations of $\mathsf{WA}^+\mathsf{WA}^a$ttacks may not result in a significant reduction in the tpr of ML-PWD, specific combinations can still have an impact on the detection performance, leading to a decrease in the tpr. The effectiveness and impact of these combinations may vary depending on the specific ML-PWD and the nature of the attacks employed.

- **MLSEC**. On the contrary, WA^r+WA^r proves to be a powerful weapon for disrupting PWD of MLSEC. As indicated in Table 3.21, the combination attack '*replOnfoc_replRet*' defeated all detectors, leading to a significant decrease in their confidence scores by 0.12–0.58. Moreover, four PWD have their confidence scores reduced below 0.5, indicating a successful evasion. Furthermore, the attack '*htEsc_replRet*' evades four detectors, resulting in a substantial reduction in their confidence scores to 0.03 or near 0.15. Additionally, the attack '*htEncd_replRet*' successfully bypasses four detectors and notably decreases the confidence score of model $m0$ from 0.91 to 0.08. These findings demonstrate the effectiveness and potency of WA^r+WA^r combination attacks in evading detection and undermining the confidence of PWD in MLSEC. The combination of multiple WA^r proves to be highly disruptive, highlighting the need for robust defense mechanisms against such attacks.

Takeaway: The simplest and cheapest attacks can indeed be highly effective in evading PWD, but their effectiveness may vary across different PWD. While these attacks may prove to be successful in bypassing certain PWD, they may not necessarily work equally well on all PWD.

Mixed: PsP and WsP

- **δPhish** and **Zenodo**. As presented in Tables 3.11, we analyze the impact of 52 attacks across the Preprocessing space and Website space of δPhish. These attacks have a detrimental effect on the detection performance of ML-PWD, particularly those based on F^r. Among these attacks, the combination attacks involving '*addHidP*' demonstrate the most significant impact on the ML-PWD. For instance, the attack '*addLngTxt_addHidP*' mentioned in Table 3.11a reduce the *tpr* of F^r-based ML-PWD from 0.79, 0.95 and 0.78 to 0.03, 0 and 0 respectively. This indicates a drastic reduction in the ability of the ML-PWD to detect and classify phishing instances. Similar situation is observed in ML-PWD of Zenodo, as illustrated in Table 3.27, the combination attack of PA^r+WA^r demonstrates a decrease in the *tpr* of ML-PWD trained on Zenodo. Notably, the attack '*delFtr_addHidP*' leads to a significant reduction in the *tpr* of F^r-based RF-PWD, dropping from 0.9 to 0.15. Furthermore, when encountering attack '*delSpn_addHidP*', the *tpr* decreases to 0.03. Other PA^r+WA^r combination attacks also prove effective in bypassing the detection of ML-PWD of Zenodo. For example, the attack '*delFtr_replPass*' results in a similar drop, and '*delFtr_addSusLnk*' reduces the *tpr* by 0.4–0.65. These findings highlight the susceptibility of ML-PWD trained on Zenodo to PA^r+WA^r attacks.

- **MLSEC**. We executed 53 kinds of PA^r+WA^r on MLSEC's PWD and evaluated their impact, which is reported in Tables 3.20. All of these combination attacks affected the decision of PWD, with 51 (i.e., except '*delSpn_modBgClr*' and '*delFtr_modBgClr*') out of 53 attacks noticeably degrading the confidence of at least one PWD. One particular attack, '*delFrm_addHidP*' minimizes the confidence of all PWD. Specifically, the confidence of $m0$ and $m2$ dropped from 0.9 to 0.01, while the confidence of other PWDs decreased by 0.16–0.5. This substantial reduction caused by this cheap attack is both shocking and expected, as this combination attack simultaneously considers the "feature space" and "problem space", i.e., both the high-level definitions of adversarial perturbations [214].

Takeaway: Comparing to other attacks mixing evasion spaces, it is evident that PA^r+WA^r possess greater destructive power and have a substantial impact on the *tpr* of PWD. These attacks are particularly potent because they traverse both the 'feature-space' (e.g., Preprocessing space) and 'problem-space' (e.g., Website space).

Mixed: PsP and MsP

- **δPhish** and **Zenodo**. We showcase 3 combination PA^r+MA^r attacks target ML-PWD trained on δPhish and Zenodo in Table 3.13 and Table 3.28, respectively. It is worth noting that these combination attacks are difficult to achieve and require high costs, as attackers must obtain write-access to deeper segments of the ML-PWD. Interestingly, despite the high cost associated with these attacks, they do not consistently and effectively disrupt ML-PWD, except for the attack '*delFtr_brTgs*' which reduces the *tpr* of F^r-based CN-PWD from 0.86 to 0.64.

- **MLSEC**. As depicted in Table 3.19, the combination attack PA^r+MA^r decreases the performance of all considered ML-PWD, but the impact is relatively minor.

The largest impact is observed with '*delSpn_brTgs*' and '*delFtr_brTgs*'. These attacks lead to a decrease in the confidence of $m0$ by 0.16 and 0.13, respectively.

Takeaway: Costly attacks (which require both MsP and PsP) do not always possess formidable attacking capabilities. They may slightly affect certain detectors or have no impact on others.

3.5 Summary

This paper aims to promote the development of secure ML systems. To do so, it is necessary to devise a realistic threat model which portrays adversarial attacks that are not only physically realizable, but also economically viable. At the same time, it is necessary to evaluate the attack's impact by crafting the corresponding perturbations. In the context of phishing detection, we formalized the evasion space to explain 'where' the perturbation can be introduced to bypass ML-based phishing website detectors. Furthermore, we proposed a realistic threat model for evasion attacks against ML-based phishing website detectors. Our threat model is grounded on detailed security considerations from the viewpoint of a typical phisher, who is confined in the 'website-space'. Nevertheless, our model can be relaxed by assuming attackers with greater capabilities but which require higher costs.

We provided lots of attack examples in different spaces, from single spaces extended to *deeper* and *mixed* spaces, to carry out a large evaluation of evasion attacks exploiting diverse 'spaces'. We focus on those requiring less resources to be staged in reality. Our paper provided a constructive step towards developing secure ML systems against adversarial attacks and paved the way for a much-needed re-assessment of adversarial attacks against ML systems for cybersecurity.

3.6 Appendix

3.6.1 Complete Benchmark Tables

We carry out our experiments by developing original software tools, all written in Python3 by leveraging well-known libraries (e.g., scikit-learn, Tensorflow). The ML-PWD using *RF* and *LR* are assessed on a system mounting an Intel Xeon W-2223@3.6GHz with 32GB RAM. For the *CN*, we use an nVidia P100 GPU.

TABLE 3.3: Evasion Robustness of the ML-PWD on the `Zenodo` dataset. The cells report the average (and std. dev.) *tpr* over the 50 reiterations. Lines correspond to the ML-PWD, while rows correspond to a specific attack.

$\mathcal{A}$	F	no-atk	WA^u	WA^r	WA^c	$\widehat{\mathsf{WA}^u}$	$\widehat{\mathsf{WA}^r}$	$\widehat{\mathsf{WA}^c}$	PA^u	PA^r	PA^c	MA^u	MA^r	MA^c
CN	F^u	$0.96_{\pm 0.007}$	$1.00_{\pm 0.000}$	$0.93_{\pm 0.020}$	$1.00_{\pm 0.000}$	$1.00_{\pm 0.000}$	$0.95_{\pm 0.018}$	$1.00_{\pm 0.000}$	$1.00_{\pm 0.017}$	$0.95_{\pm 0.018}$	$1.00_{\pm 0.017}$	$0.18_{\pm 0.222}$	$0.95_{\pm 0.018}$	$0.18_{\pm 0.222}$
	F^r	$0.86_{\pm 0.013}$	$0.88_{\pm 0.013}$	$0.87_{\pm 0.056}$	$0.87_{\pm 0.055}$	$0.88_{\pm 0.013}$	$0.44_{\pm 0.153}$	$0.83_{\pm 0.051}$	$0.54_{\pm 0.108}$	$0.29_{\pm 0.120}$	$0.31_{\pm 0.118}$	$0.88_{\pm 0.013}$	$0.02_{\pm 0.095}$	$0.02_{\pm 0.095}$
	F^c	$0.97_{\pm 0.009}$	$0.92_{\pm 0.036}$	$0.93_{\pm 0.020}$	$0.94_{\pm 0.063}$	$0.92_{\pm 0.036}$	$0.92_{\pm 0.016}$	$0.83_{\pm 0.115}$	$1.00_{\pm 0.011}$	$0.90_{\pm 0.031}$	$0.99_{\pm 0.017}$	$0.51_{\pm 0.131}$	$0.92_{\pm 0.036}$	$0.15_{\pm 0.211}$
RF	F^u	$0.96_{\pm 0.007}$	$1.00_{\pm 0.000}$	$0.96_{\pm 0.008}$	$1.00_{\pm 0.000}$	$1.00_{\pm 0.000}$	$0.96_{\pm 0.008}$	$1.00_{\pm 0.000}$	$0.54_{\pm 0.183}$	$0.96_{\pm 0.007}$	$0.54_{\pm 0.183}$	$0.04_{\pm 0.098}$	$0.96_{\pm 0.007}$	$0.04_{\pm 0.098}$
	F^r	$0.90_{\pm 0.013}$	$0.90_{\pm 0.013}$	$0.88_{\pm 0.024}$	$0.88_{\pm 0.025}$	$0.90_{\pm 0.013}$	$0.71_{\pm 0.053}$	$0.80_{\pm 0.025}$	$0.59_{\pm 0.086}$	$0.47_{\pm 0.082}$	$0.30_{\pm 0.088}$	$0.90_{\pm 0.013}$	$0.04_{\pm 0.155}$	$0.04_{\pm 0.155}$
	F^c	$0.97_{\pm 0.009}$	$0.98_{\pm 0.064}$	$0.94_{\pm 0.012}$	$0.94_{\pm 0.171}$	$0.98_{\pm 0.063}$	$0.94_{\pm 0.010}$	$0.94_{\pm 0.191}$	$0.65_{\pm 0.101}$	$0.94_{\pm 0.010}$	$0.21_{\pm 0.134}$	$0.07_{\pm 0.115}$	$0.92_{\pm 0.012}$	$0.03_{\pm 0.158}$
LR	F^u	$0.97_{\pm 0.005}$	$1.00_{\pm 0.000}$	$0.95_{\pm 0.005}$	$1.00_{\pm 0.000}$	$1.00_{\pm 0.000}$	$0.96_{\pm 0.005}$	$1.00_{\pm 0.000}$	$0.73_{\pm 0.071}$	$0.96_{\pm 0.006}$	$0.73_{\pm 0.071}$	$0.00_{\pm 0.000}$	$0.96_{\pm 0.006}$	$0.00_{\pm 0.000}$
	F^r	$0.80_{\pm 0.013}$	$0.80_{\pm 0.013}$	$0.65_{\pm 0.043}$	$0.64_{\pm 0.040}$	$0.80_{\pm 0.013}$	$0.54_{\pm 0.027}$	$0.56_{\pm 0.022}$	$0.61_{\pm 0.007}$	$0.08_{\pm 0.013}$	$0.01_{\pm 0.010}$	$0.80_{\pm 0.013}$	$0.00_{\pm 0.000}$	$0.00_{\pm 0.000}$
	F^c	$0.98_{\pm 0.005}$	$0.82_{\pm 0.035}$	$0.95_{\pm 0.015}$	$0.32_{\pm 0.079}$	$0.80_{\pm 0.038}$	$0.93_{\pm 0.014}$	$0.32_{\pm 0.132}$	$0.46_{\pm 0.053}$	$0.91_{\pm 0.032}$	$0.06_{\pm 0.025}$	$0.00_{\pm 0.000}$	$0.76_{\pm 0.036}$	$0.00_{\pm 0.000}$

TABLE 3.4: Evasion Robustness of the ML-PWD on the δphish dataset. The cells report the average (and std. dev.) *tpr* over the 50 reiterations. Lines correspond to the ML-PWD, while rows correspond to a specific attack.

$\mathcal{A}$	F	no-atk	WA^u	WA^r	WA^c	$\widehat{WA}^u$	$\widehat{WA}^r$	$\widehat{WA}^c$	PA^u	PA^r	PA^c	MA^u	MA^r	MA^c
CN	F^u	$0.65_{\pm 0.028}$	$0.91_{\pm 0.276}$	$0.65_{\pm 0.029}$	$0.91_{\pm 0.275}$	$0.90_{\pm 0.299}$	$0.65_{\pm 0.029}$	$0.90_{\pm 0.300}$	$0.60_{\pm 0.165}$	$0.65_{\pm 0.028}$	$0.60_{\pm 0.165}$	$0.14_{\pm 0.346}$	$0.65_{\pm 0.028}$	$0.14_{\pm 0.346}$
	F^r	$0.79_{\pm 0.013}$	$0.80_{\pm 0.013}$	$0.35_{\pm 0.018}$	$0.34_{\pm 0.017}$	$0.80_{\pm 0.013}$	$0.86_{\pm 0.033}$	$0.88_{\pm 0.020}$	$0.46_{\pm 0.065}$	$0.69_{\pm 0.038}$	$0.46_{\pm 0.064}$	$0.81_{\pm 0.013}$	$0.00_{\pm 0.000}$	$0.00_{\pm 0.000}$
	F^c	$0.95_{\pm 0.010}$	$0.88_{\pm 0.066}$	$0.93_{\pm 0.012}$	$0.84_{\pm 0.113}$	$0.89_{\pm 0.046}$	$0.89_{\pm 0.020}$	$0.87_{\pm 0.058}$	$0.90_{\pm 0.107}$	$0.58_{\pm 0.059}$	$0.82_{\pm 0.163}$	$0.04_{\pm 0.198}$	$0.01_{\pm 0.011}$	$0.04_{\pm 0.196}$
RF	F^u	$0.56_{\pm 0.037}$	$0.84_{\pm 0.330}$	$0.56_{\pm 0.036}$	$0.84_{\pm 0.330}$	$0.84_{\pm 0.330}$	$0.56_{\pm 0.034}$	$0.84_{\pm 0.331}$	$0.57_{\pm 0.238}$	$0.56_{\pm 0.037}$	$0.57_{\pm 0.238}$	$0.01_{\pm 0.053}$	$0.56_{\pm 0.037}$	$0.01_{\pm 0.053}$
	F^r	$0.95_{\pm 0.008}$	$0.95_{\pm 0.009}$	$0.84_{\pm 0.003}$	$0.84_{\pm 0.043}$	$0.95_{\pm 0.009}$	$0.80_{\pm 0.038}$	$0.94_{\pm 0.009}$	$0.84_{\pm 0.049}$	$0.55_{\pm 0.090}$	$0.95_{\pm 0.055}$	$0.95_{\pm 0.008}$	$0.00_{\pm 0.000}$	$0.00_{\pm 0.000}$
	F^c	$0.95_{\pm 0.009}$	$0.90_{\pm 0.020}$	$0.92_{\pm 0.006}$	$0.77_{\pm 0.047}$	$0.90_{\pm 0.017}$	$0.86_{\pm 0.018}$	$0.92_{\pm 0.015}$	$0.90_{\pm 0.065}$	$0.68_{\pm 0.013}$	$0.86_{\pm 0.097}$	$0.88_{\pm 0.026}$	$0.00_{\pm 0.001}$	$0.00_{\pm 0.000}$
LR	F^u	$0.30_{\pm 0.014}$	$0.21_{\pm 0.332}$	$0.30_{\pm 0.015}$	$0.22_{\pm 0.341}$	$0.26_{\pm 0.364}$	$0.30_{\pm 0.015}$	$0.24_{\pm 0.359}$	$0.64_{\pm 0.256}$	$0.30_{\pm 0.014}$	$0.64_{\pm 0.256}$	$0.00_{\pm 0.000}$	$0.30_{\pm 0.014}$	$0.00_{\pm 0.000}$
	F^r	$0.78_{\pm 0.011}$	$0.78_{\pm 0.011}$	$0.57_{\pm 0.014}$	$0.56_{\pm 0.047}$	$0.78_{\pm 0.011}$	$0.60_{\pm 0.030}$	$0.63_{\pm 0.010}$	$0.80_{\pm 0.029}$	$0.04_{\pm 0.006}$	$0.45_{\pm 0.068}$	$0.78_{\pm 0.011}$	$0.00_{\pm 0.000}$	$0.00_{\pm 0.000}$
	F^c	$0.86_{\pm 0.014}$	$0.47_{\pm 0.094}$	$0.81_{\pm 0.011}$	$0.36_{\pm 0.102}$	$0.73_{\pm 0.126}$	$0.73_{\pm 0.018}$	$0.63_{\pm 0.150}$	$0.65_{\pm 0.157}$	$0.23_{\pm 0.014}$	$0.32_{\pm 0.109}$	$0.00_{\pm 0.000}$	$0.00_{\pm 0.000}$	$0.00_{\pm 0.000}$

Evasion Performance We report the complete results of all the 12 considered evasion attacks against all the 18 considered ML-PWD in Table 3.3 (for Zenodo) and Table 3.4 (for δphish). These tables also include the performance in non-adversarial settings computed on the 100 phishing samples (drawn from P_i that are used as base for the adversarial samples). We remark that we chose such 100 samples by randomly selecting 100 samples which were correctly detected by the best ML-PWD on each dataset. As such, the *tpr* reported in the *no-atk* column can slightly differ from the one in Table 2.3 (which is computed on the entire P_i).

Runtime. We report in Table 3.5 the runtime for training and testing all our ML-PWD in non-adversarial scenarios. The values denote the average runtime (and standard deviation) across the 50 trials. Training the *RF* and *LR* uses all cores/threads of our CPU.

TABLE 3.5: Execution Times for training (on $\mathcal{D}$) and testing (on both P_i and B_i) the ML models used by our ML-PWD.

$\mathcal{A}$	F	Zenodo		δphish	
		Train (s)	Test (ms)	Train (s)	Test (ms)
CN	F^u	$110.88_{\pm 15.318}$	$178.13_{\pm 9.661}$	$201.314_{\pm 21.753}$	$301.91_{\pm 46.133}$
	F^r	$76.61_{\pm 4.562}$	$171.95_{\pm 10.577}$	$167.74_{\pm 25.197}$	$273.4_{\pm 43.99}$
	F^c	$152.325_{\pm 13.183}$	$222.696_{\pm 86.618}$	$165.486_{\pm 23.367}$	$274.84_{\pm 47.975}$
RF	F^u	$0.152_{\pm 0.0052}$	$7.59_{\pm 0.208}$	$0.583_{\pm 0.0181}$	$28.09_{\pm 0.402}$
	F^r	$0.146_{\pm 0.0037}$	$7.85_{\pm 0.07}$	$0.369_{\pm 0.0181}$	$22.39_{\pm 0.151}$
	F^c	$0.179_{\pm 0.0035}$	$9.39_{\pm 0.312}$	$0.44_{\pm 0.0062}$	$23.6_{\pm 0.205}$
LR	F^u	$0.045_{\pm 0.019}$	$0.1_{\pm 0.005}$	$0.185_{\pm 0.0285}$	$0.45_{\pm 0.895}$
	F^r	$0.055_{\pm 0.0182}$	$0.09_{\pm 0.003}$	$0.083_{\pm 0.0509}$	$0.74_{\pm 1.161}$
	F^c	$0.063_{\pm 0.0179}$	$0.17_{\pm 0.014}$	$0.301_{\pm 0.0678}$	$0.36_{\pm 0.678}$

3.6.2 Alternative WA^r for Zenodo and δPhish

As we mentioned in § 3.1, we applied two different WA^r to the ML-PWD of δPhish and Zenodo (i.e., replOnc: swap <a href='link'> into <a onclick="this.href='link'"> on Zenodo, and addInLnk: insert can not see to the samples of δPhish), and report their influence in Figs. 2.4. In this section,

we apply the same WA^r, but with the datasets swapped to see if the influence will change, i.e., applying addInLnk to `Zenodo` and applying replOnc to δ`Phish`. The new influence on each dataset is depicted in Table. 3.6. Comparing with the Figs. 2.4, it can be concluded that the δ`Phish` is more vulnerable to addInLnk, whereas their impact on `Zenodo` are similar.

TABLE 3.6: Impact of Alternative WA^r on ML-PWD generated on `Zenodo` and δ`Phish`, reported as the average (and std. dev.) *tpr* over the 50 trials.

$\mathcal{A}$	F	`Zenodo`		δ`phish`	
		tpr (no-atk)	*tpr* (addInLnk)	*tpr* (no-atk)	*tpr*(replOnc)
CN	F^u	$0.96_{\pm 0.008}$	$0.95_{\pm 0.018}$	$0.55_{\pm 0.030}$	$0.65_{\pm 0.029}$
	F^r	$0.88_{\pm 0.018}$	$0.61_{\pm 0.034}$	$0.81_{\pm 0.019}$	$0.89_{\pm 0.018}$
	F^c	$0.97_{\pm 0.006}$	$0.97_{\pm 0.021}$	$0.93_{\pm 0.013}$	$0.93_{\pm 0.012}$
RF	F^u	$0.98_{\pm 0.004}$	$0.96_{\pm 0.008}$	$0.45_{\pm 0.022}$	$0.56_{\pm 0.036}$
	F^r	$0.93_{\pm 0.013}$	$0.94_{\pm 0.018}$	$0.94_{\pm 0.016}$	$0.99_{\pm 0.003}$
	F^c	$0.98_{\pm 0.006}$	$0.97_{\pm 0.008}$	$0.97_{\pm 0.007}$	$0.98_{\pm 0.006}$
LR	F^u	$0.95_{\pm 0.009}$	$0.96_{\pm 0.002}$	$0.24_{\pm 0.017}$	$0.3_{\pm 0.015}$
	F^r	$0.82_{\pm 0.017}$	$0.95_{\pm 0.005}$	$0.74_{\pm 0.025}$	$0.78_{\pm 0.014}$
	F^c	$0.96_{\pm 0.007}$	$0.98_{\pm 0.007}$	$0.81_{\pm 0.020}$	$0.89_{\pm 0.011}$

3.6.3 Supplementary Tables for Additional Experiments

We now report the *complete* results of *all* our new experiments, which we discussed in §3.4.

Perturbation's impact on δ`Phish`

We report new WA^r's impact on the ML-PWD generated on δ`Phish` in Table 3.7 and Table 3.8. PsP and WsP's influence were depicted in Table 3.9. And Table 3.10 describes the *tpr* of ML-PWD generated on δ`Phish` against WA^u. Table 3.12, 3.13, 3.11 and 3.14 report the influence of hybrid space attacks on δ`Phish`.

TABLE 3.7: Evasion Robustness of the ML-PWD against iWA^r on the δ`Phish`. The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the ML-PWD, while rows correspond to a specific iWsP perturbation.

$\mathcal{A}$	F	no-atk	replOnc	delHidIt	addHidP	replJS	replRet	htEsc	htEncd	replPass	replOnfoc	addSusLnk
CN	F^u	$0.65_{\pm 0.028}$	$0.65_{\pm 0.029}$	$0.65_{\pm 0.029}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.035}$	$0.64_{\pm 0.031}$
	F^r	$0.79_{\pm 0.013}$	$0.89_{\pm 0.018}$	$0.81_{\pm 0.013}$	$0.03_{\pm 0.006}$	$0.79_{\pm 0.011}$	$0.81_{\pm 0.013}$	$0.94_{\pm 0.03}$	$1.0_{\pm 0.0}$	$0.81_{\pm 0.013}$	$0.81_{\pm 0.013}$	$0.19_{\pm 0.012}$
	F^c	$0.95_{\pm 0.010}$	$0.93_{\pm 0.012}$	$0.95_{\pm 0.016}$	$0.22_{\pm 0.059}$	$0.89_{\pm 0.021}$	$0.96_{\pm 0.011}$	$0.99_{\pm 0.01}$	$0.99_{\pm 0.014}$	$0.95_{\pm 0.011}$	$0.95_{\pm 0.013}$	$0.79_{\pm 0.039}$
RF	F^u	$0.56_{\pm 0.037}$	$0.56_{\pm 0.036}$	$0.56_{\pm 0.035}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.034}$	$0.57_{\pm 0.033}$	$0.57_{\pm 0.031}$	$0.56_{\pm 0.033}$	$0.57_{\pm 0.033}$	$0.56_{\pm 0.037}$	$0.56_{\pm 0.032}$
	F^r	$0.95_{\pm 0.008}$	$0.99_{\pm 0.003}$	$0.88_{\pm 0.011}$	$0.0_{\pm 0.0}$	$0.81_{\pm 0.021}$	$0.95_{\pm 0.008}$	$1.0_{\pm 0.003}$	$1.0_{\pm 0.0}$	$0.95_{\pm 0.008}$	$0.95_{\pm 0.008}$	$0.44_{\pm 0.069}$
	F^c	$0.95_{\pm 0.009}$	$0.98_{\pm 0.006}$	$0.93_{\pm 0.01}$	$0.04_{\pm 0.017}$	$0.86_{\pm 0.015}$	$0.95_{\pm 0.01}$	$1.0_{\pm 0.007}$	$1.0_{\pm 0.0}$	$0.95_{\pm 0.009}$	$0.94_{\pm 0.009}$	$0.48_{\pm 0.043}$
LR	F^u	$0.30_{\pm 0.014}$	$0.3_{\pm 0.015}$	$0.29_{\pm 0.015}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.014}$	$0.3_{\pm 0.014}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.014}$	$0.3_{\pm 0.021}$	$0.3_{\pm 0.014}$
	F^r	$0.78_{\pm 0.011}$	$0.78_{\pm 0.014}$	$0.68_{\pm 0.017}$	$0.0_{\pm 0.0}$	$0.68_{\pm 0.005}$	$0.78_{\pm 0.011}$	$0.84_{\pm 0.006}$	$1.0_{\pm 0.0}$	$0.78_{\pm 0.011}$	$0.78_{\pm 0.011}$	$0.3_{\pm 0.009}$
	F^c	$0.86_{\pm 0.014}$	$0.89_{\pm 0.011}$	$0.82_{\pm 0.016}$	$0.17_{\pm 0.015}$	$0.78_{\pm 0.01}$	$0.86_{\pm 0.014}$	$0.92_{\pm 0.015}$	$1.0_{\pm 0.005}$	$0.87_{\pm 0.014}$	$0.74_{\pm 0.042}$	$0.62_{\pm 0.025}$

TABLE 3.8: Evasion Robustness of the ML-PWD against eWA^r and rWA^r on the δPhish. The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the ML-PWD, while rows correspond to a specific eWsP or rWsP attack.

$\mathcal{A}$	F	no-atk	eWsP								rWsP			
			addImgBot	modFntTyp	modCpy	addIcn	delSusLnk	delSusFrm	modTtl	delCpy	modBgimg	modBgClr	modFntClr	modFntSiz
CN	F^u	$0.65_{\pm 0.028}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.63_{\pm 0.036}$	$0.64_{\pm 0.031}$
	F^r	$0.79_{\pm 0.013}$	$0.63_{\pm 0.063}$	$0.81_{\pm 0.013}$	$0.77_{\pm 0.016}$	$0.71_{\pm 0.024}$	$0.84_{\pm 0.021}$	$0.75_{\pm 0.012}$	$0.81_{\pm 0.013}$	$0.77_{\pm 0.016}$	$0.81_{\pm 0.013}$	$0.81_{\pm 0.013}$	$0.81_{\pm 0.013}$	$0.81_{\pm 0.013}$
	F^c	$0.95_{\pm 0.010}$	$0.92_{\pm 0.032}$	$0.95_{\pm 0.011}$	$0.94_{\pm 0.014}$	$0.92_{\pm 0.021}$	$0.93_{\pm 0.012}$	$0.93_{\pm 0.016}$	$0.95_{\pm 0.011}$	$0.94_{\pm 0.014}$	$0.95_{\pm 0.011}$	$0.95_{\pm 0.011}$	$0.94_{\pm 0.017}$	$0.95_{\pm 0.011}$
RF	F^u	$0.56_{\pm 0.037}$	$0.57_{\pm 0.034}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.032}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.034}$	$0.56_{\pm 0.034}$	$0.57_{\pm 0.034}$	$0.56_{\pm 0.036}$	$0.56_{\pm 0.033}$
	F^r	$0.95_{\pm 0.008}$	$0.88_{\pm 0.026}$	$0.95_{\pm 0.008}$	$0.95_{\pm 0.007}$	$0.89_{\pm 0.019}$	$0.92_{\pm 0.011}$	$0.91_{\pm 0.021}$	$0.95_{\pm 0.008}$	$0.95_{\pm 0.007}$	$0.95_{\pm 0.008}$	$0.95_{\pm 0.008}$	$0.95_{\pm 0.008}$	$0.95_{\pm 0.008}$
	F^c	$0.95_{\pm 0.009}$	$0.88_{\pm 0.015}$	$0.95_{\pm 0.009}$	$0.94_{\pm 0.009}$	$0.89_{\pm 0.015}$	$0.92_{\pm 0.007}$	$0.91_{\pm 0.009}$	$0.95_{\pm 0.009}$	$0.94_{\pm 0.009}$	$0.95_{\pm 0.009}$	$0.95_{\pm 0.009}$	$0.94_{\pm 0.009}$	$0.95_{\pm 0.009}$
LR	F^u	$0.30_{\pm 0.014}$	$0.3_{\pm 0.014}$	$0.3_{\pm 0.014}$	$0.3_{\pm 0.014}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.014}$	$0.3_{\pm 0.024}$	$0.3_{\pm 0.014}$
	F^r	$0.78_{\pm 0.011}$	$0.47_{\pm 0.026}$	$0.78_{\pm 0.011}$	$0.77_{\pm 0.011}$	$0.61_{\pm 0.015}$	$0.83_{\pm 0.007}$	$0.75_{\pm 0.025}$	$0.79_{\pm 0.011}$	$0.77_{\pm 0.011}$	$0.78_{\pm 0.011}$	$0.78_{\pm 0.011}$	$0.78_{\pm 0.011}$	$0.78_{\pm 0.011}$
	F^c	$0.86_{\pm 0.014}$	$0.66_{\pm 0.028}$	$0.87_{\pm 0.014}$	$0.89_{\pm 0.013}$	$0.82_{\pm 0.013}$	$0.91_{\pm 0.009}$	$0.78_{\pm 0.018}$	$0.87_{\pm 0.014}$	$0.89_{\pm 0.013}$	$0.87_{\pm 0.013}$	$0.87_{\pm 0.014}$	$0.74_{\pm 0.044}$	$0.87_{\pm 0.014}$

TABLE 3.9: Impact of PA^r and MA^r on ML-PWD generated on δphish. The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the ML-PWD, while rows correspond to a specific PsP or MsP attack.

$\mathcal{A}$	F	no-atk	PsP							MsP					
			delTxt	delFrm	delSpn	delTtl	addLngTxt	delFtr	replSusFtrLnk	brTg	delHt	delHd	delBdy	brTgs	hmg
CN	F^u	$0.65_{\pm 0.028}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.65_{\pm 0.032}$
	F^r	$0.79_{\pm 0.013}$	$0.78_{\pm 0.014}$	$0.75_{\pm 0.012}$	$0.8_{\pm 0.013}$	$0.81_{\pm 0.013}$	$0.81_{\pm 0.013}$	$0.76_{\pm 0.015}$	$0.79_{\pm 0.011}$	$0.81_{\pm 0.013}$	$1.0_{\pm 0.0}$	$0.79_{\pm 0.009}$	$0.87_{\pm 0.018}$	$0.81_{\pm 0.012}$	$0.76_{\pm 0.019}$
	F^c	$0.95_{\pm 0.010}$	$0.89_{\pm 0.034}$	$0.93_{\pm 0.016}$	$0.95_{\pm 0.012}$	$0.91_{\pm 0.027}$	$0.95_{\pm 0.011}$	$0.93_{\pm 0.013}$	$0.95_{\pm 0.011}$	$0.95_{\pm 0.011}$	$0.99_{\pm 0.014}$	$0.82_{\pm 0.045}$	$0.98_{\pm 0.015}$	$0.95_{\pm 0.011}$	$0.78_{\pm 0.034}$
RF	F^u	$0.56_{\pm 0.037}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.032}$	$0.57_{\pm 0.032}$	$0.57_{\pm 0.032}$	$0.57_{\pm 0.033}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.035}$	$0.56_{\pm 0.035}$	$0.56_{\pm 0.035}$	$0.56_{\pm 0.033}$	$0.57_{\pm 0.033}$	$0.56_{\pm 0.034}$	$0.56_{\pm 0.036}$
	F^r	$0.95_{\pm 0.008}$	$0.94_{\pm 0.012}$	$0.91_{\pm 0.021}$	$0.95_{\pm 0.007}$	$0.94_{\pm 0.012}$	$0.95_{\pm 0.008}$	$0.91_{\pm 0.01}$	$0.94_{\pm 0.011}$	$0.95_{\pm 0.008}$	$1.0_{\pm 0.0}$	$0.83_{\pm 0.019}$	$1.0_{\pm 0.003}$	$0.95_{\pm 0.008}$	$0.79_{\pm 0.024}$
	F^c	$0.95_{\pm 0.009}$	$0.92_{\pm 0.012}$	$0.91_{\pm 0.009}$	$0.94_{\pm 0.009}$	$0.93_{\pm 0.011}$	$0.95_{\pm 0.009}$	$0.94_{\pm 0.01}$	$0.94_{\pm 0.009}$	$0.95_{\pm 0.009}$	$1.0_{\pm 0.0}$	$0.86_{\pm 0.015}$	$1.0_{\pm 0.007}$	$0.94_{\pm 0.009}$	$0.8_{\pm 0.017}$
LR	F^u	$0.30_{\pm 0.014}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.014}$	$0.3_{\pm 0.014}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.014}$
	F^r	$0.78_{\pm 0.011}$	$0.64_{\pm 0.024}$	$0.75_{\pm 0.025}$	$0.75_{\pm 0.016}$	$0.64_{\pm 0.025}$	$0.78_{\pm 0.011}$	$0.79_{\pm 0.016}$	$0.76_{\pm 0.011}$	$0.78_{\pm 0.011}$	$1.0_{\pm 0.0}$	$0.65_{\pm 0.01}$	$0.84_{\pm 0.006}$	$0.78_{\pm 0.011}$	$0.69_{\pm 0.01}$
	F^c	$0.86_{\pm 0.014}$	$0.78_{\pm 0.018}$	$0.78_{\pm 0.018}$	$0.87_{\pm 0.014}$	$0.76_{\pm 0.02}$	$0.87_{\pm 0.014}$	$0.89_{\pm 0.014}$	$0.85_{\pm 0.012}$	$0.87_{\pm 0.013}$	$1.0_{\pm 0.004}$	$0.76_{\pm 0.03}$	$0.95_{\pm 0.008}$	$0.87_{\pm 0.013}$	$0.76_{\pm 0.013}$

TABLE 3.10: Impact of WA^u on ML-PWD of δPhish.

$\mathcal{A}$	F	no-atk	replChar	sepWrd	delChar	swpChar	addChar	atkPth
CN	F^u	$0.65_{\pm 0.028}$	$0.64_{\pm 0.043}$	$0.64_{\pm 0.038}$	$0.63_{\pm 0.033}$	$0.63_{\pm 0.037}$	$0.64_{\pm 0.044}$	$0.6_{\pm 0.029}$
	F^r	$0.79_{\pm 0.013}$	$0.81_{\pm 0.013}$	$0.79_{\pm 0.016}$	$0.8_{\pm 0.014}$	$0.81_{\pm 0.014}$	$0.81_{\pm 0.014}$	$0.8_{\pm 0.013}$
	F^c	$0.95_{\pm 0.010}$	$0.95_{\pm 0.009}$	$0.95_{\pm 0.01}$	$0.94_{\pm 0.01}$	$0.95_{\pm 0.011}$	$0.94_{\pm 0.012}$	$0.94_{\pm 0.009}$
RF	F^u	$0.56_{\pm 0.037}$	$0.56_{\pm 0.03}$	$0.59_{\pm 0.024}$	$0.56_{\pm 0.029}$	$0.56_{\pm 0.032}$	$0.56_{\pm 0.031}$	$0.52_{\pm 0.027}$
	F^r	$0.95_{\pm 0.008}$	$0.95_{\pm 0.009}$	$0.95_{\pm 0.008}$	$0.95_{\pm 0.009}$	$0.95_{\pm 0.009}$	$0.95_{\pm 0.009}$	$0.95_{\pm 0.008}$
	F^c	$0.95_{\pm 0.009}$	$0.94_{\pm 0.009}$	$0.94_{\pm 0.009}$	$0.92_{\pm 0.011}$	$0.94_{\pm 0.009}$	$0.94_{\pm 0.009}$	$0.94_{\pm 0.01}$
LR	F^u	$0.30_{\pm 0.014}$	$0.3_{\pm 0.02}$	$0.31_{\pm 0.024}$	$0.28_{\pm 0.019}$	$0.28_{\pm 0.02}$	$0.29_{\pm 0.019}$	$0.29_{\pm 0.015}$
	F^r	$0.78_{\pm 0.011}$	$0.78_{\pm 0.011}$	$0.79_{\pm 0.012}$	$0.77_{\pm 0.012}$	$0.78_{\pm 0.012}$	$0.78_{\pm 0.011}$	$0.78_{\pm 0.011}$
	F^c	$0.86_{\pm 0.014}$	$0.83_{\pm 0.018}$	$0.85_{\pm 0.028}$	$0.84_{\pm 0.016}$	$0.83_{\pm 0.019}$	$0.83_{\pm 0.021}$	$0.88_{\pm 0.01}$

TABLE 3.11: Impact of PA^r+WA^r on ML-PWD generated on δphish. The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the ML-PWD, while rows correspond to a specific PsP+WsP perturbation.

(A) PA^r+WA^r

$\mathcal{A}$	F	no-atk	addLngTxt_ addInLnk	addLngTxt_ delHidIt	addLngTxt_ replOnfoc	addLngTxt_ replPass	addLngTxt_ addHidP	addLngTxt_ replJS	addLngTxt_ delSusLnk	addLngTxt_ modCpy	addLngTxt_ modTtl	addLngTxt_ addIcn	addLngTxt_ addSusLnk	addLngTxt_ addImgBot	addLngTxt_ modBgClr
CN	F^u	$0.65_{\pm 0.028}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.032}$	$0.64_{\pm 0.031}$	$0.63_{\pm 0.036}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$
	F^r	$0.79_{\pm 0.013}$	$0.35_{\pm 0.017}$	$0.81_{\pm 0.013}$	$0.81_{\pm 0.013}$	$0.81_{\pm 0.013}$	$0.03_{\pm 0.006}$	$0.79_{\pm 0.011}$	$0.84_{\pm 0.021}$	$0.77_{\pm 0.016}$	$0.81_{\pm 0.013}$	$0.71_{\pm 0.024}$	$0.2_{\pm 0.013}$	$0.63_{\pm 0.063}$	$0.81_{\pm 0.013}$
	F^c	$0.95_{\pm 0.010}$	$0.92_{\pm 0.029}$	$0.95_{\pm 0.016}$	$0.95_{\pm 0.011}$	$0.95_{\pm 0.011}$	$0.22_{\pm 0.039}$	$0.89_{\pm 0.021}$	$0.92_{\pm 0.017}$	$0.94_{\pm 0.014}$	$0.95_{\pm 0.011}$	$0.92_{\pm 0.021}$	$0.8_{\pm 0.033}$	$0.91_{\pm 0.035}$	$0.95_{\pm 0.011}$
RF	F^u	$0.56_{\pm 0.037}$	$0.56_{\pm 0.035}$	$0.57_{\pm 0.034}$	$0.56_{\pm 0.033}$	$0.57_{\pm 0.031}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.032}$	$0.56_{\pm 0.035}$	$0.56_{\pm 0.034}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.034}$	$0.56_{\pm 0.034}$	$0.56_{\pm 0.034}$	$0.56_{\pm 0.034}$
	F^r	$0.95_{\pm 0.008}$	$0.84_{\pm 0.043}$	$0.88_{\pm 0.011}$	$0.95_{\pm 0.008}$	$0.95_{\pm 0.008}$	$0.0_{\pm 0.0}$	$0.81_{\pm 0.021}$	$0.92_{\pm 0.011}$	$0.95_{\pm 0.007}$	$0.95_{\pm 0.008}$	$0.89_{\pm 0.019}$	$0.46_{\pm 0.045}$	$0.88_{\pm 0.026}$	$0.95_{\pm 0.008}$
	F^c	$0.95_{\pm 0.009}$	$0.92_{\pm 0.017}$	$0.93_{\pm 0.01}$	$0.95_{\pm 0.009}$	$0.95_{\pm 0.009}$	$0.04_{\pm 0.017}$	$0.86_{\pm 0.015}$	$0.92_{\pm 0.008}$	$0.94_{\pm 0.009}$	$0.95_{\pm 0.009}$	$0.89_{\pm 0.015}$	$0.58_{\pm 0.039}$	$0.88_{\pm 0.015}$	$0.95_{\pm 0.009}$
LR	F^u	$0.30_{\pm 0.014}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.014}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.024}$	$0.3_{\pm 0.014}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.014}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.016}$
	F^r	$0.78_{\pm 0.011}$	$0.57_{\pm 0.045}$	$0.68_{\pm 0.017}$	$0.78_{\pm 0.011}$	$0.78_{\pm 0.011}$	$0.0_{\pm 0.0}$	$0.68_{\pm 0.005}$	$0.83_{\pm 0.007}$	$0.77_{\pm 0.011}$	$0.79_{\pm 0.011}$	$0.61_{\pm 0.015}$	$0.38_{\pm 0.026}$	$0.44_{\pm 0.026}$	$0.78_{\pm 0.011}$
	F^c	$0.86_{\pm 0.014}$	$0.8_{\pm 0.017}$	$0.81_{\pm 0.014}$	$0.87_{\pm 0.013}$	$0.87_{\pm 0.014}$	$0.17_{\pm 0.016}$	$0.78_{\pm 0.01}$	$0.8_{\pm 0.043}$	$0.89_{\pm 0.013}$	$0.87_{\pm 0.014}$	$0.82_{\pm 0.012}$	$0.68_{\pm 0.021}$	$0.66_{\pm 0.03}$	$0.87_{\pm 0.015}$

(B) PA^r+WA^r

$\mathcal{A}$	F	no-atk	delFrm_ addInLnk	delFrm_ onclick	delFrm_ delHidIt	delFrm_ addHidP	delFrm_ replJS	delFrm_ delSusLnk	delFrm_ addImgBot	delFrm_ modFntSiz	delFrm_ modBgimg	delFrm_ modBgClr	delFrm_ delCpy	delFrm_ modTtl	delFrm_ modCpy	delFrm_ addIcn	delFrm_ replRet
CN	F^u	$0.65_{\pm 0.028}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$
	F^r	$0.79_{\pm 0.013}$	$0.25_{\pm 0.021}$	$0.93_{\pm 0.019}$	$0.76_{\pm 0.011}$	$0.01_{\pm 0.004}$	$0.73_{\pm 0.009}$	$0.87_{\pm 0.018}$	$0.54_{\pm 0.081}$	$0.76_{\pm 0.011}$	$0.76_{\pm 0.011}$	$0.76_{\pm 0.011}$	$0.72_{\pm 0.014}$	$0.76_{\pm 0.011}$	$0.72_{\pm 0.014}$	$0.64_{\pm 0.024}$	$0.76_{\pm 0.011}$
	F^c	$0.95_{\pm 0.010}$	$0.85_{\pm 0.04}$	$0.97_{\pm 0.014}$	$0.93_{\pm 0.017}$	$0.19_{\pm 0.062}$	$0.81_{\pm 0.021}$	$0.91_{\pm 0.013}$	$0.88_{\pm 0.036}$	$0.91_{\pm 0.017}$	$0.91_{\pm 0.017}$	$0.91_{\pm 0.017}$	$0.89_{\pm 0.02}$	$0.91_{\pm 0.017}$	$0.89_{\pm 0.02}$	$0.86_{\pm 0.031}$	$0.91_{\pm 0.017}$
RF	F^u	$0.56_{\pm 0.037}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.034}$	$0.56_{\pm 0.034}$	$0.57_{\pm 0.032}$	$0.57_{\pm 0.033}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.034}$	$0.57_{\pm 0.032}$	$0.56_{\pm 0.034}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.034}$	$0.57_{\pm 0.032}$	$0.56_{\pm 0.036}$
	F^r	$0.95_{\pm 0.008}$	$0.85_{\pm 0.055}$	$0.99_{\pm 0.007}$	$0.89_{\pm 0.015}$	$0.0_{\pm 0.0}$	$0.74_{\pm 0.015}$	$0.91_{\pm 0.014}$	$0.85_{\pm 0.018}$	$0.9_{\pm 0.011}$	$0.9_{\pm 0.011}$	$0.9_{\pm 0.011}$	$0.9_{\pm 0.01}$	$0.9_{\pm 0.011}$	$0.9_{\pm 0.01}$	$0.82_{\pm 0.02}$	$0.9_{\pm 0.011}$
	F^c	$0.95_{\pm 0.009}$	$0.88_{\pm 0.023}$	$0.98_{\pm 0.011}$	$0.92_{\pm 0.009}$	$0.03_{\pm 0.016}$	$0.8_{\pm 0.011}$	$0.91_{\pm 0.011}$	$0.87_{\pm 0.017}$	$0.9_{\pm 0.012}$	$0.9_{\pm 0.012}$	$0.9_{\pm 0.012}$	$0.91_{\pm 0.009}$	$0.9_{\pm 0.012}$	$0.91_{\pm 0.009}$	$0.83_{\pm 0.021}$	$0.9_{\pm 0.012}$
LR	F^u	$0.30_{\pm 0.014}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.014}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.014}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.017}$
	F^r	$0.78_{\pm 0.011}$	$0.43_{\pm 0.082}$	$0.79_{\pm 0.013}$	$0.72_{\pm 0.024}$	$0.0_{\pm 0.0}$	$0.68_{\pm 0.01}$	$0.84_{\pm 0.008}$	$0.43_{\pm 0.04}$	$0.74_{\pm 0.019}$	$0.74_{\pm 0.019}$	$0.74_{\pm 0.019}$	$0.73_{\pm 0.019}$	$0.75_{\pm 0.019}$	$0.73_{\pm 0.019}$	$0.57_{\pm 0.022}$	$0.74_{\pm 0.019}$
	F^c	$0.86_{\pm 0.014}$	$0.7_{\pm 0.028}$	$0.85_{\pm 0.019}$	$0.82_{\pm 0.012}$	$0.16_{\pm 0.011}$	$0.73_{\pm 0.01}$	$0.89_{\pm 0.013}$	$0.57_{\pm 0.026}$	$0.79_{\pm 0.012}$	$0.79_{\pm 0.011}$	$0.79_{\pm 0.011}$	$0.81_{\pm 0.016}$	$0.79_{\pm 0.011}$	$0.81_{\pm 0.016}$	$0.74_{\pm 0.013}$	$0.79_{\pm 0.011}$

(C) PA^r+WA^r

$\mathcal{A}$	F	no-atk	delFtr_ addInLnk	delFtr_ delHidIt	delFtr_ replOnfoc	delFtr_ replPass	delFtr_ addHidP	delFtr_ replJS	delFtr_ delSusLnk	delFtr_ modCpy	delFtr_ modTtl	delFtr_ addIcn	delFtr_ addSusLnk	delFtr_ modBgClr
CN	F^u	$0.65_{\pm 0.028}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$
	F^r	$0.79_{\pm 0.013}$	$0.26_{\pm 0.012}$	$0.76_{\pm 0.014}$	$0.76_{\pm 0.015}$	$0.76_{\pm 0.015}$	$0.01_{\pm 0.01}$	$0.76_{\pm 0.014}$	$0.83_{\pm 0.021}$	$0.75_{\pm 0.017}$	$0.76_{\pm 0.015}$	$0.64_{\pm 0.022}$	$0.14_{\pm 0.015}$	$0.76_{\pm 0.015}$
	F^c	$0.95_{\pm 0.010}$	$0.9_{\pm 0.031}$	$0.92_{\pm 0.019}$	$0.93_{\pm 0.013}$	$0.93_{\pm 0.013}$	$0.22_{\pm 0.058}$	$0.87_{\pm 0.019}$	$0.92_{\pm 0.011}$	$0.92_{\pm 0.013}$	$0.93_{\pm 0.013}$	$0.88_{\pm 0.019}$	$0.77_{\pm 0.032}$	$0.93_{\pm 0.013}$
RF	F^u	$0.56_{\pm 0.037}$	$0.57_{\pm 0.032}$	$0.56_{\pm 0.034}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.034}$	$0.56_{\pm 0.032}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.034}$	$0.56_{\pm 0.034}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.034}$	$0.56_{\pm 0.033}$
	F^r	$0.95_{\pm 0.008}$	$0.86_{\pm 0.044}$	$0.86_{\pm 0.016}$	$0.91_{\pm 0.01}$	$0.91_{\pm 0.01}$	$0.0_{\pm 0.001}$	$0.77_{\pm 0.021}$	$0.92_{\pm 0.009}$	$0.91_{\pm 0.009}$	$0.91_{\pm 0.01}$	$0.83_{\pm 0.015}$	$0.44_{\pm 0.055}$	$0.91_{\pm 0.01}$
	F^c	$0.95_{\pm 0.009}$	$0.91_{\pm 0.021}$	$0.91_{\pm 0.009}$	$0.94_{\pm 0.01}$	$0.94_{\pm 0.01}$	$0.03_{\pm 0.016}$	$0.85_{\pm 0.012}$	$0.92_{\pm 0.008}$	$0.93_{\pm 0.012}$	$0.94_{\pm 0.01}$	$0.86_{\pm 0.021}$	$0.56_{\pm 0.045}$	$0.94_{\pm 0.01}$
LR	F^u	$0.30_{\pm 0.014}$	$0.3_{\pm 0.014}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.014}$
	F^r	$0.78_{\pm 0.011}$	$0.61_{\pm 0.053}$	$0.69_{\pm 0.014}$	$0.79_{\pm 0.016}$	$0.79_{\pm 0.016}$	$0.0_{\pm 0.0}$	$0.71_{\pm 0.012}$	$0.85_{\pm 0.003}$	$0.78_{\pm 0.016}$	$0.8_{\pm 0.016}$	$0.62_{\pm 0.016}$	$0.38_{\pm 0.028}$	$0.79_{\pm 0.016}$
	F^c	$0.86_{\pm 0.014}$	$0.84_{\pm 0.016}$	$0.82_{\pm 0.012}$	$0.89_{\pm 0.014}$	$0.89_{\pm 0.014}$	$0.19_{\pm 0.021}$	$0.78_{\pm 0.01}$	$0.9_{\pm 0.009}$	$0.89_{\pm 0.012}$	$0.89_{\pm 0.014}$	$0.85_{\pm 0.014}$	$0.72_{\pm 0.019}$	$0.89_{\pm 0.014}$

(D) PA^r+WA^r

$\mathcal{A}$	F	no-atk	delSpn_ addInLnk	delSpn_ replOnfoc	delSpn_ replPass	delSpn_ addHidP	delSpn_ replJS	delSpn_ delSusLnk	delSpn_ modCpy	delSpn_ modTtl	delSpn_ addIcn	delSpn_ addSusLnk	delSpn_ addImgBot	delSpn_ modBgClr
CN	F^u	$0.65_{\pm 0.028}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$	$0.64_{\pm 0.031}$
	F^r	$0.79_{\pm 0.013}$	$0.35_{\pm 0.019}$	$0.8_{\pm 0.013}$	$0.8_{\pm 0.013}$	$0.06_{\pm 0.025}$	$0.78_{\pm 0.011}$	$0.82_{\pm 0.019}$	$0.78_{\pm 0.014}$	$0.8_{\pm 0.013}$	$0.69_{\pm 0.022}$	$0.22_{\pm 0.023}$	$0.61_{\pm 0.067}$	$0.8_{\pm 0.013}$
	F^c	$0.95_{\pm 0.010}$	$0.92_{\pm 0.033}$	$0.95_{\pm 0.012}$	$0.95_{\pm 0.012}$	$0.22_{\pm 0.059}$	$0.89_{\pm 0.021}$	$0.93_{\pm 0.013}$	$0.94_{\pm 0.014}$	$0.95_{\pm 0.012}$	$0.92_{\pm 0.022}$	$0.8_{\pm 0.035}$	$0.89_{\pm 0.042}$	$0.95_{\pm 0.012}$
RF	F^u	$0.56_{\pm 0.037}$	$0.57_{\pm 0.032}$	$0.56_{\pm 0.032}$	$0.57_{\pm 0.033}$	$0.56_{\pm 0.032}$	$0.56_{\pm 0.034}$	$0.56_{\pm 0.034}$	$0.56_{\pm 0.033}$	$0.56_{\pm 0.032}$	$0.57_{\pm 0.035}$	$0.56_{\pm 0.032}$	$0.56_{\pm 0.032}$	$0.57_{\pm 0.032}$
	F^r	$0.95_{\pm 0.008}$	$0.84_{\pm 0.042}$	$0.95_{\pm 0.007}$	$0.95_{\pm 0.007}$	$0.0_{\pm 0.0}$	$0.84_{\pm 0.024}$	$0.92_{\pm 0.011}$	$0.95_{\pm 0.006}$	$0.95_{\pm 0.007}$	$0.89_{\pm 0.019}$	$0.45_{\pm 0.045}$	$0.9_{\pm 0.015}$	$0.95_{\pm 0.007}$
	F^c	$0.95_{\pm 0.009}$	$0.92_{\pm 0.015}$	$0.94_{\pm 0.009}$	$0.94_{\pm 0.009}$	$0.03_{\pm 0.016}$	$0.85_{\pm 0.015}$	$0.92_{\pm 0.007}$	$0.94_{\pm 0.009}$	$0.94_{\pm 0.009}$	$0.89_{\pm 0.015}$	$0.54_{\pm 0.036}$	$0.88_{\pm 0.015}$	$0.94_{\pm 0.009}$
LR	F^u	$0.30_{\pm 0.014}$	$0.3_{\pm 0.014}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.016}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.014}$	$0.3_{\pm 0.015}$	$0.3_{\pm 0.014}$
	F^r	$0.78_{\pm 0.011}$	$0.55_{\pm 0.056}$	$0.75_{\pm 0.016}$	$0.75_{\pm 0.016}$	$0.0_{\pm 0.0}$	$0.68_{\pm 0.009}$	$0.83_{\pm 0.007}$	$0.74_{\pm 0.016}$	$0.76_{\pm 0.016}$	$0.6_{\pm 0.028}$	$0.32_{\pm 0.018}$	$0.45_{\pm 0.028}$	$0.75_{\pm 0.016}$
	F^c	$0.86_{\pm 0.014}$	$0.8_{\pm 0.019}$	$0.87_{\pm 0.014}$	$0.87_{\pm 0.014}$	$0.17_{\pm 0.014}$	$0.79_{\pm 0.009}$	$0.91_{\pm 0.008}$	$0.89_{\pm 0.014}$	$0.87_{\pm 0.014}$	$0.82_{\pm 0.013}$	$0.66_{\pm 0.024}$	$0.65_{\pm 0.027}$	$0.87_{\pm 0.014}$

TABLE 3.12: Impact of PA^r+PA^r on ML-PWD generated on δPhish.The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the ML-PWD, while rows correspond to a specific PsP+PsP perturbation.

$\mathcal{A}$	F	no-atk	addLngTxt_delTtl	delFtr_delTtl	delFtr_addLngTxt	delSpn_delTtl	delSpn_delFtrer	delSpn_addLngTxt	delFrm_delFtr	delFrm_delSpn
CN	F^u	0.65±0.028	0.64±0.031	0.64±0.031	0.64±0.031	0.64±0.031	0.64±0.031	0.64±0.031	0.64±0.031	0.64±0.031
	F^r	0.79±0.013	0.81±0.013	0.76±0.015	0.76±0.015	0.8±0.013	0.75±0.015	0.8±0.013	1.0±0.0	0.74±0.01
	F^c	0.95±0.010	0.91±0.027	0.89±0.026	0.93±0.013	0.9±0.03	0.92±0.014	0.95±0.012	0.99±0.014	0.9±0.018
RF	F^u	0.56±0.037	0.56±0.033	0.57±0.033	0.56±0.032	0.56±0.033	0.57±0.033	0.56±0.034	0.56±0.034	0.56±0.032
	F^r	0.95±0.008	0.94±0.012	0.89±0.014	0.91±0.01	0.94±0.011	0.91±0.011	0.95±0.007	1.0±0.0	0.91±0.01
	F^c	0.95±0.009	0.93±0.011	0.91±0.016	0.94±0.01	0.92±0.012	0.94±0.01	0.94±0.009	1.0±0.0	0.92±0.008
LR	F^u	0.30±0.014	0.3±0.015	0.3±0.015	0.3±0.014	0.3±0.015	0.3±0.015	0.3±0.014	0.3±0.015	0.3±0.014
	F^r	0.78±0.011	0.64±0.025	0.64±0.024	0.79±0.016	0.62±0.035	0.76±0.02	0.75±0.016	1.0±0.0	0.74±0.019
	F^c	0.86±0.014	0.76±0.02	0.79±0.019	0.89±0.014	0.76±0.021	0.89±0.014	0.87±0.014	1.0±0.005	0.81±0.01

TABLE 3.13: Impact of PA^r+MA^r attacks on δPhish.The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the ML-PWD, while rows correspond to a specific PsP+MsP.

$\mathcal{A}$	F	no-atk	addLngTxt_delBdy	delfoot_delBdy	delSpn_delBdy
CN	F^u	0.65±0.028	0.64±0.031	0.64±0.031	0.64±0.031
	F^r	0.79±0.013	0.81±0.012	0.76±0.015	0.8±0.013
	F^c	0.95±0.010	0.95±0.011	0.93±0.013	0.95±0.012
RF	F^u	0.56±0.037	0.56±0.031	0.56±0.034	0.56±0.033
	F^r	0.95±0.008	0.95±0.008	0.91±0.01	0.95±0.006
	F^c	0.95±0.009	0.94±0.009	0.94±0.01	0.94±0.009
LR	F^u	0.30±0.014	0.3±0.014	0.3±0.014	0.3±0.015
	F^r	0.78±0.011	0.78±0.011	0.79±0.016	0.75±0.016
	F^c	0.86±0.014	0.87±0.014	0.89±0.014	0.87±0.014

TABLE 3.14: Impact of WA^r+WA^r on δPhish.The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the ML-PWD, while rows correspond to a specific WsP+WsP.

$\mathcal{A}$	F	no-atk	replOnfoc_replRet	htEsc_replRet	htEncd_replRet
CN	F^u	0.65±0.028	0.64±0.031	0.63±0.035	0.64±0.031
	F^r	0.79±0.013	0.81±0.013	1.0±0.0	1.0±0.0
	F^c	0.95±0.010	0.96±0.011	0.97±0.033	0.99±0.014
RF	F^u	0.56±0.037	0.56±0.034	0.56±0.036	0.56±0.032
	F^r	0.95±0.008	0.95±0.008	1.0±0.0	1.0±0.0
	F^c	0.95±0.009	0.95±0.01	1.0±0.0	1.0±0.0
LR	F^u	0.30±0.014	0.3±0.016	0.3±0.023	0.3±0.015
	F^r	0.78±0.011	0.78±0.011	1.0±0.0	1.0±0.0
	F^c	0.86±0.014	0.86±0.014	0.91±0.069	1.0±0.004

Perturbation's impact on MLSEC

We executed 37 kinds of single attacks and report the influence of MLSEC's PWD in Table 3.15, 3.16 and 3.17, and the influence of hybrid space attacks in Table 3.19, 3.18, 3.20 and 3.21.

TABLE 3.15: Impact of iWA^r on the PWD of MLSEC. The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the ML-PWD, while rows correspond to a specific iWsP perturbation.

$\mathcal{A}$	no-atk	addInLnk	replOnc	delHidIt	addHidP	replJS	replRet	htEsc	htEncd	replPass	replOnfoc	addSusLnk
$m0$	0.91±0.052	0.73±0.159	0.91±0.052	0.88±0.049	0.02±0.011	0.9±0.059	0.79±0.123	0.06±0.046	0.06±0.036	0.48±0.259	0.5±0.254	0.5±0.203
$m1$	0.87±0.071	0.86±0.083	0.88±0.07	0.85±0.093	0.52±0.161	0.85±0.088	0.74±0.115	0.37±0.113	0.41±0.126	0.85±0.068	0.84±0.09	0.85±0.1
$m2$	0.9±0.051	0.73±0.158	0.9±0.052	0.88±0.052	0.02±0.011	0.9±0.058	0.83±0.105	0.85±0.127	0.9±0.051	0.47±0.266	0.51±0.263	0.5±0.202
$m3$	0.88±0.07	0.86±0.08	0.87±0.07	0.85±0.096	0.51±0.172	0.85±0.087	0.79±0.108	0.86±0.099	0.88±0.07	0.85±0.066	0.85±0.093	0.85±0.1
$m4$	0.82±0.106	0.83±0.108	0.82±0.111	0.8±0.136	0.83±0.123	0.82±0.126	0.69±0.122	0.09±0.067	0.07±0.045	0.46±0.256	0.46±0.242	0.47±0.171
$m5$	0.81±0.12	0.82±0.12	0.81±0.124	0.8±0.141	0.82±0.136	0.81±0.136	0.67±0.114	0.43±0.098	0.46±0.139	0.79±0.125	0.79±0.119	0.8±0.157
$m6$	0.83±0.108	0.83±0.11	0.83±0.112	0.81±0.131	0.84±0.116	0.82±0.127	0.73±0.148	0.84±0.126	0.83±0.108	0.47±0.256	0.49±0.236	0.47±0.174
$m7$	0.82±0.121	0.82±0.122	0.82±0.126	0.81±0.136	0.83±0.127	0.81±0.138	0.7±0.145	0.84±0.121	0.82±0.121	0.79±0.126	0.81±0.125	0.8±0.157

TABLE 3.16: Evasion Robustness of the MLSEC's PWD against eWA^r and rWA^r. The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the PWD, while rows correspond to a specific eWsP or rWsP attack.

$\mathcal{A}$	no-atk	eWsP								rWsP			
		addImgBot	modFntTyp	modCpy	addIcn	delSusLnk	delSusFrm	modTtl	delCpy	modBgimg	modBgClr	modFntClr	modFntSiz
$m0$	0.91±0.052	0.65±0.185	0.92±0.051	0.89±0.061	0.76±0.126	0.86±0.095	0.8±0.253	0.87±0.102	0.89±0.061	0.92±0.053	0.91±0.056	0.81±0.116	0.92±0.051
$m1$	0.87±0.071	0.87±0.085	0.89±0.063	0.85±0.091	0.77±0.089	0.82±0.146	0.81±0.106	0.84±0.12	0.85±0.089	0.89±0.064	0.88±0.077	0.78±0.1	0.89±0.063
$m2$	0.9±0.051	0.65±0.185	0.91±0.05	0.89±0.06	0.76±0.122	0.86±0.095	0.79±0.262	0.87±0.101	0.89±0.06	0.91±0.052	0.91±0.055	0.85±0.087	0.91±0.05
$m3$	0.88±0.07	0.87±0.079	0.89±0.064	0.85±0.09	0.77±0.081	0.82±0.146	0.8±0.124	0.84±0.119	0.85±0.088	0.89±0.066	0.88±0.076	0.81±0.091	0.89±0.064
$m4$	0.82±0.106	0.64±0.199	0.87±0.065	0.81±0.124	0.83±0.109	0.8±0.156	0.73±0.257	0.8±0.156	0.81±0.127	0.87±0.066	0.86±0.079	0.73±0.112	0.87±0.065
$m5$	0.81±0.12	0.85±0.107	0.85±0.089	0.8±0.136	0.82±0.122	0.79±0.145	0.78±0.14	0.79±0.167	0.8±0.137	0.85±0.089	0.84±0.096	0.7±0.103	0.85±0.089
$m6$	0.83±0.108	0.64±0.198	0.87±0.066	0.81±0.126	0.83±0.109	0.8±0.157	0.72±0.261	0.79±0.157	0.81±0.128	0.87±0.066	0.86±0.079	0.73±0.111	0.87±0.065
$m7$	0.82±0.121	0.85±0.106	0.85±0.089	0.8±0.138	0.82±0.123	0.79±0.146	0.78±0.141	0.79±0.169	0.81±0.138	0.85±0.089	0.84±0.097	0.7±0.097	0.85±0.089

TABLE 3.17: Impact of PA^r and MA^r on PWD of MLSEC. The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the ML-PWD, while rows correspond to a specific PsP or MsP attack.

$\mathcal{A}$	no-atk	PsP							MsP					
		delTxt	delFrm	delSpn	delTtl	addLngTxt	delFtr	replSusFtrLnk	brTg	delHt	delHd	delBdy	brTgs	hmg
$m0$	0.91±0.052	0.64±0.272	0.91±0.052	0.85±0.089	0.88±0.062	0.86±0.095	0.87±0.085	0.9±0.059	0.9±0.062	0.9±0.062	0.9±0.062	0.9±0.062	0.9±0.062	0.9±0.062
$m1$	0.87±0.071	0.83±0.133	0.67±0.262	0.67±0.262	0.82±0.117	0.83±0.133	0.82±0.117	0.67±0.262	0.31±0.051	0.31±0.051	0.31±0.051	0.31±0.051	0.31±0.051	0.31±0.051
$m2$	0.9±0.051	0.84±0.148	0.61±0.39	0.61±0.39	0.85±0.087	0.84±0.148	0.85±0.087	0.61±0.39	0.88±0.096	0.88±0.096	0.88±0.096	0.88±0.096	0.88±0.096	0.88±0.096
$m3$	0.88±0.07	0.83±0.131	0.66±0.271	0.66±0.271	0.82±0.115	0.83±0.131	0.82±0.115	0.66±0.271	0.26±0.08	0.26±0.08	0.26±0.08	0.26±0.08	0.26±0.08	0.26±0.08
$m4$	0.82±0.106	0.8±0.169	0.57±0.372	0.57±0.372	0.79±0.149	0.8±0.169	0.79±0.149	0.57±0.372	0.8±0.121	0.8±0.121	0.8±0.121	0.8±0.121	0.8±0.121	0.8±0.121
$m5$	0.81±0.12	0.79±0.16	0.64±0.28	0.64±0.28	0.79±0.143	0.79±0.16	0.79±0.143	0.64±0.28	0.39±0.166	0.39±0.166	0.39±0.166	0.39±0.166	0.39±0.166	0.39±0.166
$m6$	0.83±0.108	0.8±0.17	0.56±0.373	0.56±0.373	0.79±0.144	0.8±0.17	0.79±0.144	0.56±0.373	0.07±0.076	0.07±0.076	0.07±0.076	0.07±0.076	0.07±0.076	0.07±0.076
$m7$	0.82±0.121	0.79±0.161	0.64±0.279	0.64±0.279	0.79±0.138	0.79±0.161	0.79±0.138	0.64±0.279	0.36±0.129	0.36±0.129	0.36±0.129	0.36±0.129	0.36±0.129	0.36±0.129

TABLE 3.18: Impact of PA^r+PA^r on MLSEC.The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the PWD, while rows correspond to a specific PsP+PsP perturbation.

$\mathcal{A}$	no-atk	addLngTxt_delTtl	delFtr_delTtl	delFtr_addLngTxt	delSpn_delTtl	delSpn_delFtrer	delSpn_addLngTxt	delFrm_delFtr	delFrm_delSpn
$m0$	0.91±0.052	0.82±0.16	0.85±0.092	0.8±0.167	0.84±0.093	0.84±0.096	0.79±0.173	0.59±0.375	0.58±0.375
$m1$	0.87±0.071	0.81±0.145	0.84±0.089	0.81±0.13	0.81±0.117	0.83±0.105	0.79±0.157	0.66±0.269	0.62±0.28
$m2$	0.9±0.051	0.81±0.159	0.85±0.094	0.8±0.166	0.83±0.095	0.84±0.096	0.79±0.172	0.59±0.376	0.58±0.376
$m3$	0.88±0.07	0.81±0.143	0.83±0.093	0.81±0.128	0.81±0.118	0.83±0.106	0.79±0.154	0.65±0.276	0.62±0.288
$m4$	0.82±0.106	0.77±0.178	0.78±0.146	0.75±0.2	0.78±0.145	0.77±0.156	0.76±0.204	0.57±0.358	0.56±0.367
$m5$	0.81±0.12	0.77±0.172	0.8±0.111	0.77±0.157	0.78±0.141	0.8±0.124	0.77±0.191	0.66±0.259	0.63±0.275
$m6$	0.83±0.108	0.77±0.177	0.77±0.155	0.74±0.198	0.77±0.152	0.77±0.154	0.76±0.201	0.56±0.358	0.56±0.367
$m7$	0.82±0.121	0.77±0.17	0.8±0.12	0.77±0.155	0.78±0.146	0.8±0.122	0.77±0.186	0.67±0.255	0.63±0.273

TABLE 3.19: Impact of PA^r+MA^r on MLSEC. The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the PWD, while rows correspond to a specific PsP+MsP perturbation.

$\mathcal{A}$	no-atk	addLngTxt_delBdy	delfoot_delBdy	delSpn_delBdy
$m0$	0.91±0.052	0.78±0.167	0.79±0.137	0.75±0.156
$m1$	0.87±0.071	0.8±0.146	0.81±0.115	0.78±0.14
$m2$	0.9±0.051	0.8±0.161	0.8±0.13	0.77±0.149
$m3$	0.88±0.07	0.83±0.125	0.82±0.108	0.79±0.133
$m4$	0.82±0.106	0.79±0.136	0.76±0.154	0.72±0.184
$m5$	0.81±0.12	0.8±0.139	0.8±0.122	0.77±0.148
$m6$	0.83±0.108	0.79±0.137	0.76±0.154	0.72±0.183
$m7$	0.82±0.121	0.8±0.139	0.8±0.123	0.77±0.145

Perturbation's impact on `Zenodo`

In this section, we present new perturbation's influence on `Zenodo`, single attacks' influence is shown in Table 3.22, 3.23, 3.25, and hybrid attacks' impact is shown in Table 3.28, 3.26, 3.27 3.29.

TABLE 3.20: Impact of PA^r+WA^r on PWD of MLSEC. The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the PWD, while rows correspond to a specific PsP+WsP perturbation.

(A) PA^r+WA^r

$\mathcal{A}$	no-atk	delFrm_ addInLnk	delFrm_ onclick	delFrm_ delHidIt	delFrm_ addHidP	delFrm_ replJS	delFrm_ delSusLnk	delFrm_ addImgBot	delFrm_ modFntSiz	delFrm_ modBgimg	delFrm_ modBgClr	delFrm_ delCpy	delFrm_ modTtl	delFrm_ modCpy	delFrm_ addIcn	delFrm_ replRet	delFrm_ combine
$m0$	$0.91_{\pm 0.052}$	$0.49_{\pm 0.335}$	$0.61_{\pm 0.388}$	$0.6_{\pm 0.384}$	$0.01_{\pm 0.008}$	$0.61_{\pm 0.388}$	$0.59_{\pm 0.379}$	$0.43_{\pm 0.325}$	$0.63_{\pm 0.388}$	$0.63_{\pm 0.388}$	$0.63_{\pm 0.387}$	$0.61_{\pm 0.387}$	$0.59_{\pm 0.383}$	$0.61_{\pm 0.386}$	$0.49_{\pm 0.329}$	$0.51_{\pm 0.341}$	$0.01_{\pm 0.008}$
$m1$	$0.87_{\pm 0.071}$	$0.66_{\pm 0.268}$	$0.67_{\pm 0.262}$	$0.63_{\pm 0.284}$	$0.41_{\pm 0.181}$	$0.67_{\pm 0.262}$	$0.62_{\pm 0.291}$	$0.69_{\pm 0.243}$	$0.69_{\pm 0.265}$	$0.69_{\pm 0.265}$	$0.69_{\pm 0.265}$	$0.67_{\pm 0.261}$	$0.65_{\pm 0.27}$	$0.66_{\pm 0.261}$	$0.55_{\pm 0.254}$	$0.53_{\pm 0.257}$	$0.37_{\pm 0.196}$
$m2$	$0.9_{\pm 0.051}$	$0.49_{\pm 0.338}$	$0.61_{\pm 0.39}$	$0.61_{\pm 0.386}$	$0.01_{\pm 0.008}$	$0.61_{\pm 0.39}$	$0.59_{\pm 0.38}$	$0.43_{\pm 0.326}$	$0.63_{\pm 0.388}$	$0.63_{\pm 0.388}$	$0.63_{\pm 0.387}$	$0.61_{\pm 0.389}$	$0.59_{\pm 0.385}$	$0.61_{\pm 0.388}$	$0.49_{\pm 0.33}$	$0.57_{\pm 0.36}$	$0.01_{\pm 0.008}$
$m3$	$0.88_{\pm 0.07}$	$0.66_{\pm 0.277}$	$0.66_{\pm 0.271}$	$0.63_{\pm 0.289}$	$0.38_{\pm 0.196}$	$0.66_{\pm 0.271}$	$0.61_{\pm 0.299}$	$0.68_{\pm 0.253}$	$0.69_{\pm 0.266}$	$0.69_{\pm 0.266}$	$0.69_{\pm 0.266}$	$0.66_{\pm 0.269}$	$0.64_{\pm 0.278}$	$0.66_{\pm 0.269}$	$0.55_{\pm 0.257}$	$0.6_{\pm 0.253}$	$0.37_{\pm 0.198}$
$m4$	$0.82_{\pm 0.106}$	$0.57_{\pm 0.372}$	$0.56_{\pm 0.372}$	$0.55_{\pm 0.378}$	$0.57_{\pm 0.379}$	$0.56_{\pm 0.372}$	$0.56_{\pm 0.378}$	$0.42_{\pm 0.324}$	$0.6_{\pm 0.377}$	$0.6_{\pm 0.377}$	$0.6_{\pm 0.377}$	$0.57_{\pm 0.366}$	$0.55_{\pm 0.375}$	$0.56_{\pm 0.371}$	$0.56_{\pm 0.372}$	$0.47_{\pm 0.314}$	$0.54_{\pm 0.388}$
$m5$	$0.81_{\pm 0.12}$	$0.64_{\pm 0.28}$	$0.63_{\pm 0.28}$	$0.61_{\pm 0.3}$	$0.65_{\pm 0.28}$	$0.64_{\pm 0.28}$	$0.62_{\pm 0.293}$	$0.68_{\pm 0.241}$	$0.67_{\pm 0.277}$	$0.67_{\pm 0.277}$	$0.67_{\pm 0.277}$	$0.65_{\pm 0.261}$	$0.61_{\pm 0.298}$	$0.63_{\pm 0.279}$	$0.63_{\pm 0.281}$	$0.5_{\pm 0.257}$	$0.6_{\pm 0.319}$
$m6$	$0.83_{\pm 0.108}$	$0.56_{\pm 0.373}$	$0.56_{\pm 0.372}$	$0.55_{\pm 0.378}$	$0.57_{\pm 0.379}$	$0.56_{\pm 0.373}$	$0.56_{\pm 0.377}$	$0.42_{\pm 0.328}$	$0.6_{\pm 0.376}$	$0.6_{\pm 0.376}$	$0.6_{\pm 0.376}$	$0.57_{\pm 0.367}$	$0.55_{\pm 0.376}$	$0.56_{\pm 0.371}$	$0.56_{\pm 0.372}$	$0.47_{\pm 0.314}$	$0.55_{\pm 0.392}$
$m7$	$0.82_{\pm 0.121}$	$0.63_{\pm 0.28}$	$0.64_{\pm 0.279}$	$0.62_{\pm 0.294}$	$0.66_{\pm 0.267}$	$0.64_{\pm 0.279}$	$0.63_{\pm 0.282}$	$0.69_{\pm 0.237}$	$0.67_{\pm 0.276}$	$0.67_{\pm 0.276}$	$0.67_{\pm 0.276}$	$0.65_{\pm 0.26}$	$0.62_{\pm 0.297}$	$0.64_{\pm 0.277}$	$0.63_{\pm 0.279}$	$0.51_{\pm 0.254}$	$0.62_{\pm 0.303}$

(B) PA^r+WA^r

$\mathcal{A}$	no-atk	delSpn_ addInLnk	delSpn_ replOnfoc	delSpn_ replPass	delSpn_ addHidP	delSpn_ replJS	delSpn_ delSusLnk	delSpn_ modCpy	delSpn_ modTtl	delSpn_ addIcn	delSpn_ addSusLnk	delSpn_ addImgBot	delSpn_ modBgClr	delSpn_ combine
$m0$	$0.91_{\pm 0.052}$	$0.73_{\pm 0.137}$	$0.41_{\pm 0.28}$	$0.39_{\pm 0.277}$	$0.02_{\pm 0.007}$	$0.85_{\pm 0.087}$	$0.84_{\pm 0.117}$	$0.85_{\pm 0.091}$	$0.83_{\pm 0.12}$	$0.7_{\pm 0.127}$	$0.68_{\pm 0.195}$	$0.6_{\pm 0.199}$	$0.88_{\pm 0.075}$	$0.0_{\pm 0.004}$
$m1$	$0.87_{\pm 0.071}$	$0.83_{\pm 0.112}$	$0.78_{\pm 0.129}$	$0.8_{\pm 0.116}$	$0.52_{\pm 0.149}$	$0.82_{\pm 0.115}$	$0.79_{\pm 0.165}$	$0.82_{\pm 0.119}$	$0.8_{\pm 0.14}$	$0.7_{\pm 0.1}$	$0.83_{\pm 0.117}$	$0.85_{\pm 0.097}$	$0.86_{\pm 0.096}$	$0.55_{\pm 0.106}$
$m2$	$0.9_{\pm 0.051}$	$0.73_{\pm 0.131}$	$0.42_{\pm 0.287}$	$0.38_{\pm 0.282}$	$0.02_{\pm 0.008}$	$0.85_{\pm 0.085}$	$0.84_{\pm 0.117}$	$0.85_{\pm 0.089}$	$0.83_{\pm 0.118}$	$0.7_{\pm 0.125}$	$0.68_{\pm 0.194}$	$0.59_{\pm 0.199}$	$0.88_{\pm 0.073}$	$0.0_{\pm 0.003}$
$m3$	$0.88_{\pm 0.07}$	$0.83_{\pm 0.104}$	$0.79_{\pm 0.129}$	$0.79_{\pm 0.111}$	$0.51_{\pm 0.17}$	$0.82_{\pm 0.113}$	$0.79_{\pm 0.164}$	$0.82_{\pm 0.117}$	$0.8_{\pm 0.137}$	$0.7_{\pm 0.097}$	$0.82_{\pm 0.114}$	$0.84_{\pm 0.094}$	$0.86_{\pm 0.094}$	$0.52_{\pm 0.125}$
$m4$	$0.82_{\pm 0.106}$	$0.79_{\pm 0.149}$	$0.43_{\pm 0.248}$	$0.43_{\pm 0.261}$	$0.8_{\pm 0.149}$	$0.79_{\pm 0.141}$	$0.77_{\pm 0.168}$	$0.78_{\pm 0.162}$	$0.76_{\pm 0.179}$	$0.78_{\pm 0.153}$	$0.65_{\pm 0.175}$	$0.6_{\pm 0.215}$	$0.84_{\pm 0.088}$	$0.5_{\pm 0.263}$
$m5$	$0.81_{\pm 0.12}$	$0.79_{\pm 0.143}$	$0.77_{\pm 0.137}$	$0.76_{\pm 0.151}$	$0.81_{\pm 0.142}$	$0.8_{\pm 0.137}$	$0.77_{\pm 0.142}$	$0.79_{\pm 0.153}$	$0.77_{\pm 0.181}$	$0.79_{\pm 0.149}$	$0.8_{\pm 0.145}$	$0.84_{\pm 0.116}$	$0.84_{\pm 0.097}$	$0.84_{\pm 0.112}$
$m6$	$0.83_{\pm 0.108}$	$0.78_{\pm 0.152}$	$0.44_{\pm 0.253}$	$0.43_{\pm 0.262}$	$0.8_{\pm 0.15}$	$0.79_{\pm 0.147}$	$0.77_{\pm 0.167}$	$0.78_{\pm 0.162}$	$0.76_{\pm 0.176}$	$0.79_{\pm 0.148}$	$0.66_{\pm 0.178}$	$0.6_{\pm 0.215}$	$0.84_{\pm 0.09}$	$0.53_{\pm 0.275}$
$m7$	$0.82_{\pm 0.121}$	$0.79_{\pm 0.148}$	$0.78_{\pm 0.136}$	$0.76_{\pm 0.145}$	$0.81_{\pm 0.142}$	$0.79_{\pm 0.14}$	$0.77_{\pm 0.14}$	$0.78_{\pm 0.152}$	$0.77_{\pm 0.177}$	$0.79_{\pm 0.144}$	$0.79_{\pm 0.143}$	$0.84_{\pm 0.113}$	$0.83_{\pm 0.098}$	$0.83_{\pm 0.116}$

(C) PA^r+WA^r

$\mathcal{A}$	no-atk	delFtr_ addInLnk	delFtr_ delHidIt	delFtr_ replOnfoc	delFtr_ replPass	delFtr_ addHidP	delFtr_ replJS	delFtr_ delSusLnk	delFtr_ modCpy	delFtr_ modTtl	delFtr_ addicon	delFtr_ addSusLnk	delFtr_ addImgBot	delFtr_ modBgClr	delFtr_ combine
$m0$	$0.91_{\pm 0.052}$	$0.77_{\pm 0.132}$	$0.77_{\pm 0.241}$	$0.49_{\pm 0.197}$	$0.51_{\pm 0.196}$	$0.02_{\pm 0.009}$	$0.87_{\pm 0.085}$	$0.84_{\pm 0.118}$	$0.87_{\pm 0.085}$	$0.85_{\pm 0.099}$	$0.72_{\pm 0.148}$	$0.71_{\pm 0.194}$	$0.67_{\pm 0.187}$	$0.89_{\pm 0.071}$	$0.0_{\pm 0.002}$
$m1$	$0.87_{\pm 0.071}$	$0.85_{\pm 0.092}$	$0.78_{\pm 0.165}$	$0.81_{\pm 0.086}$	$0.83_{\pm 0.089}$	$0.53_{\pm 0.148}$	$0.85_{\pm 0.086}$	$0.81_{\pm 0.156}$	$0.85_{\pm 0.086}$	$0.84_{\pm 0.092}$	$0.73_{\pm 0.093}$	$0.85_{\pm 0.099}$	$0.87_{\pm 0.077}$	$0.87_{\pm 0.076}$	$0.36_{\pm 0.096}$
$m2$	$0.9_{\pm 0.051}$	$0.77_{\pm 0.131}$	$0.77_{\pm 0.253}$	$0.48_{\pm 0.214}$	$0.49_{\pm 0.215}$	$0.02_{\pm 0.011}$	$0.87_{\pm 0.084}$	$0.84_{\pm 0.117}$	$0.87_{\pm 0.084}$	$0.85_{\pm 0.098}$	$0.72_{\pm 0.147}$	$0.71_{\pm 0.192}$	$0.67_{\pm 0.187}$	$0.89_{\pm 0.07}$	$0.0_{\pm 0.001}$
$m3$	$0.88_{\pm 0.07}$	$0.85_{\pm 0.092}$	$0.78_{\pm 0.18}$	$0.82_{\pm 0.094}$	$0.82_{\pm 0.091}$	$0.51_{\pm 0.173}$	$0.85_{\pm 0.085}$	$0.81_{\pm 0.155}$	$0.85_{\pm 0.085}$	$0.84_{\pm 0.09}$	$0.73_{\pm 0.092}$	$0.85_{\pm 0.096}$	$0.87_{\pm 0.076}$	$0.87_{\pm 0.075}$	$0.33_{\pm 0.11}$
$m4$	$0.82_{\pm 0.106}$	$0.79_{\pm 0.154}$	$0.71_{\pm 0.263}$	$0.5_{\pm 0.187}$	$0.54_{\pm 0.174}$	$0.81_{\pm 0.145}$	$0.79_{\pm 0.143}$	$0.77_{\pm 0.171}$	$0.79_{\pm 0.143}$	$0.79_{\pm 0.15}$	$0.79_{\pm 0.154}$	$0.67_{\pm 0.177}$	$0.65_{\pm 0.191}$	$0.85_{\pm 0.072}$	$0.57_{\pm 0.193}$
$m5$	$0.81_{\pm 0.12}$	$0.81_{\pm 0.117}$	$0.78_{\pm 0.164}$	$0.77_{\pm 0.137}$	$0.78_{\pm 0.138}$	$0.83_{\pm 0.109}$	$0.82_{\pm 0.107}$	$0.79_{\pm 0.134}$	$0.82_{\pm 0.107}$	$0.81_{\pm 0.115}$	$0.81_{\pm 0.117}$	$0.8_{\pm 0.146}$	$0.86_{\pm 0.084}$	$0.86_{\pm 0.069}$	$0.84_{\pm 0.116}$
$m6$	$0.83_{\pm 0.108}$	$0.79_{\pm 0.153}$	$0.71_{\pm 0.263}$	$0.5_{\pm 0.206}$	$0.53_{\pm 0.196}$	$0.81_{\pm 0.146}$	$0.78_{\pm 0.153}$	$0.77_{\pm 0.171}$	$0.78_{\pm 0.153}$	$0.78_{\pm 0.147}$	$0.78_{\pm 0.153}$	$0.67_{\pm 0.184}$	$0.65_{\pm 0.194}$	$0.85_{\pm 0.077}$	$0.59_{\pm 0.22}$
$m7$	$0.82_{\pm 0.121}$	$0.81_{\pm 0.116}$	$0.78_{\pm 0.156}$	$0.78_{\pm 0.142}$	$0.78_{\pm 0.136}$	$0.84_{\pm 0.109}$	$0.81_{\pm 0.116}$	$0.79_{\pm 0.134}$	$0.81_{\pm 0.116}$	$0.81_{\pm 0.112}$	$0.81_{\pm 0.116}$	$0.8_{\pm 0.148}$	$0.86_{\pm 0.084}$	$0.85_{\pm 0.075}$	$0.84_{\pm 0.118}$

(D) PA^r+WA^r

$\mathcal{A}$	no-atk	addLngTxt_ addInLnk	addLngTxt_ delHidIt	addLngTxt_ replOnfoc	addLngTxt_ replPass	addLngTxt_ addHidP	addLngTxt_ replJS	addLngTxt_ delSusLnk	addLngTxt_ modCpy	addLngTxt_ modTtl	addLngTxt_ addIcn	addLngTxt_ addSusLnk	addLngTxt_ addImgBot	addLngTxt_ modBgClr	addLngTxt_ combine
$m0$	$0.91_{\pm 0.052}$	$0.69_{\pm 0.214}$	$0.73_{\pm 0.277}$	$0.41_{\pm 0.245}$	$0.42_{\pm 0.272}$	$0.02_{\pm 0.011}$	$0.83_{\pm 0.158}$	$0.68_{\pm 0.216}$	$0.83_{\pm 0.158}$	$0.83_{\pm 0.158}$	$0.67_{\pm 0.187}$	$0.68_{\pm 0.218}$	$0.59_{\pm 0.227}$	$0.84_{\pm 0.157}$	$0.0_{\pm 0.004}$
$m1$	$0.87_{\pm 0.071}$	$0.85_{\pm 0.105}$	$0.73_{\pm 0.216}$	$0.8_{\pm 0.144}$	$0.82_{\pm 0.116}$	$0.5_{\pm 0.189}$	$0.82_{\pm 0.141}$	$0.86_{\pm 0.103}$	$0.82_{\pm 0.141}$	$0.82_{\pm 0.141}$	$0.69_{\pm 0.136}$	$0.86_{\pm 0.103}$	$0.84_{\pm 0.126}$	$0.84_{\pm 0.131}$	$0.48_{\pm 0.139}$
$m2$	$0.9_{\pm 0.051}$	$0.69_{\pm 0.213}$	$0.73_{\pm 0.281}$	$0.42_{\pm 0.25}$	$0.41_{\pm 0.275}$	$0.02_{\pm 0.011}$	$0.83_{\pm 0.157}$	$0.68_{\pm 0.215}$	$0.83_{\pm 0.157}$	$0.83_{\pm 0.157}$	$0.66_{\pm 0.189}$	$0.68_{\pm 0.215}$	$0.58_{\pm 0.227}$	$0.84_{\pm 0.156}$	$0.0_{\pm 0.004}$
$m3$	$0.88_{\pm 0.07}$	$0.86_{\pm 0.102}$	$0.73_{\pm 0.228}$	$0.81_{\pm 0.149}$	$0.82_{\pm 0.114}$	$0.5_{\pm 0.186}$	$0.82_{\pm 0.14}$	$0.86_{\pm 0.101}$	$0.82_{\pm 0.14}$	$0.82_{\pm 0.14}$	$0.69_{\pm 0.136}$	$0.86_{\pm 0.101}$	$0.83_{\pm 0.125}$	$0.84_{\pm 0.13}$	$0.48_{\pm 0.152}$
$m4$	$0.82_{\pm 0.106}$	$0.8_{\pm 0.169}$	$0.68_{\pm 0.3}$	$0.43_{\pm 0.256}$	$0.46_{\pm 0.26}$	$0.81_{\pm 0.166}$	$0.78_{\pm 0.18}$	$0.72_{\pm 0.126}$	$0.79_{\pm 0.179}$	$0.79_{\pm 0.179}$	$0.8_{\pm 0.169}$	$0.72_{\pm 0.126}$	$0.59_{\pm 0.231}$	$0.84_{\pm 0.116}$	$0.42_{\pm 0.252}$
$m5$	$0.81_{\pm 0.12}$	$0.79_{\pm 0.16}$	$0.71_{\pm 0.252}$	$0.77_{\pm 0.152}$	$0.77_{\pm 0.158}$	$0.81_{\pm 0.156}$	$0.78_{\pm 0.173}$	$0.82_{\pm 0.125}$	$0.79_{\pm 0.17}$	$0.79_{\pm 0.17}$	$0.79_{\pm 0.16}$	$0.82_{\pm 0.125}$	$0.82_{\pm 0.134}$	$0.82_{\pm 0.122}$	$0.69_{\pm 0.251}$
$m6$	$0.83_{\pm 0.108}$	$0.8_{\pm 0.169}$	$0.68_{\pm 0.3}$	$0.45_{\pm 0.238}$	$0.46_{\pm 0.263}$	$0.81_{\pm 0.167}$	$0.78_{\pm 0.179}$	$0.72_{\pm 0.128}$	$0.79_{\pm 0.178}$	$0.79_{\pm 0.178}$	$0.8_{\pm 0.17}$	$0.72_{\pm 0.128}$	$0.59_{\pm 0.23}$	$0.83_{\pm 0.115}$	$0.48_{\pm 0.259}$
$m7$	$0.82_{\pm 0.121}$	$0.79_{\pm 0.161}$	$0.71_{\pm 0.248}$	$0.78_{\pm 0.163}$	$0.77_{\pm 0.159}$	$0.81_{\pm 0.159}$	$0.78_{\pm 0.172}$	$0.82_{\pm 0.126}$	$0.79_{\pm 0.169}$	$0.79_{\pm 0.169}$	$0.8_{\pm 0.162}$	$0.82_{\pm 0.126}$	$0.82_{\pm 0.134}$	$0.82_{\pm 0.119}$	$0.72_{\pm 0.245}$

TABLE 3.21: Impact of WA^r+WA^r on MLSEC. The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the PWD, while rows correspond to a specific WsP+WsP.

$\mathcal{A}$	no-atk	replOnfoc_replRet	htEsc_replRet	htEncd_replRet
$m0$	$0.91_{\pm 0.052}$	$0.33_{\pm 0.165}$	$0.03_{\pm 0.021}$	$0.08_{\pm 0.04}$
$m1$	$0.87_{\pm 0.071}$	$0.63_{\pm 0.136}$	$0.16_{\pm 0.065}$	$0.22_{\pm 0.082}$
$m2$	$0.9_{\pm 0.051}$	$0.46_{\pm 0.229}$	$0.86_{\pm 0.137}$	$0.9_{\pm 0.051}$
$m3$	$0.88_{\pm 0.07}$	$0.76_{\pm 0.13}$	$0.86_{\pm 0.105}$	$0.88_{\pm 0.07}$
$m4$	$0.82_{\pm 0.106}$	$0.31_{\pm 0.176}$	$0.03_{\pm 0.017}$	$0.11_{\pm 0.05}$
$m5$	$0.81_{\pm 0.12}$	$0.55_{\pm 0.124}$	$0.15_{\pm 0.053}$	$0.28_{\pm 0.093}$
$m6$	$0.83_{\pm 0.108}$	$0.36_{\pm 0.221}$	$0.85_{\pm 0.124}$	$0.83_{\pm 0.108}$
$m7$	$0.82_{\pm 0.121}$	$0.62_{\pm 0.182}$	$0.85_{\pm 0.109}$	$0.82_{\pm 0.121}$

TABLE 3.22: Evasion Robustness of the ML-PWD against iWA^r on the `Zenodo`. The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the ML-PWD, while rows correspond to a specific iWsP perturbation.

$\mathcal{A}$	F	no-atk	addInLnk	delHidIt	addHidP	replJS	replRet	htEsc	htEncd	replPass	replOnfoc	addSusLnk
CN	F^u	$0.96_{\pm 0.007}$	$0.95_{\pm 0.018}$	$0.95_{\pm 0.018}$	$0.93_{\pm 0.017}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.92_{\pm 0.023}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.95_{\pm 0.018}$	$0.96_{\pm 0.013}$
	F^r	$0.86_{\pm 0.013}$	$0.61_{\pm 0.034}$	$0.88_{\pm 0.012}$	$0.28_{\pm 0.008}$	$0.74_{\pm 0.05}$	$0.88_{\pm 0.013}$	$0.87_{\pm 0.025}$	$0.0_{\pm 0.0}$	$0.88_{\pm 0.013}$	$0.88_{\pm 0.013}$	$0.48_{\pm 0.022}$
	F^c	$0.97_{\pm 0.009}$	$0.97_{\pm 0.021}$	$0.96_{\pm 0.016}$	$0.86_{\pm 0.027}$	$0.95_{\pm 0.013}$	$0.97_{\pm 0.012}$	$0.92_{\pm 0.019}$	$0.9_{\pm 0.033}$	$0.97_{\pm 0.012}$	$0.97_{\pm 0.012}$	$0.97_{\pm 0.021}$
RF	F^u	$0.96_{\pm 0.007}$	$0.96_{\pm 0.008}$	$0.96_{\pm 0.008}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.96_{\pm 0.007}$	$0.98_{\pm 0.005}$
	F^r	$0.90_{\pm 0.013}$	$0.94_{\pm 0.018}$	$0.84_{\pm 0.01}$	$0.03_{\pm 0.064}$	$0.71_{\pm 0.016}$	$0.9_{\pm 0.013}$	$0.84_{\pm 0.027}$	$0.0_{\pm 0.0}$	$0.9_{\pm 0.013}$	$0.9_{\pm 0.013}$	$0.64_{\pm 0.062}$
	F^c	$0.97_{\pm 0.009}$	$0.97_{\pm 0.008}$	$0.97_{\pm 0.01}$	$0.96_{\pm 0.006}$	$0.96_{\pm 0.006}$	$0.98_{\pm 0.004}$	$0.96_{\pm 0.007}$	$0.96_{\pm 0.007}$	$0.98_{\pm 0.005}$	$0.97_{\pm 0.01}$	$0.97_{\pm 0.008}$
LR	F^u	$0.97_{\pm 0.005}$	$0.96_{\pm 0.002}$	$0.96_{\pm 0.005}$	$0.97_{\pm 0.005}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.005}$	$0.97_{\pm 0.009}$	$0.97_{\pm 0.004}$	$0.95_{\pm 0.005}$	$0.97_{\pm 0.004}$
	F^r	$0.80_{\pm 0.013}$	$0.95_{\pm 0.005}$	$0.79_{\pm 0.014}$	$0.24_{\pm 0.019}$	$0.46_{\pm 0.013}$	$0.8_{\pm 0.013}$	$0.55_{\pm 0.009}$	$0.0_{\pm 0.0}$	$0.8_{\pm 0.013}$	$0.8_{\pm 0.013}$	$0.72_{\pm 0.0}$
	F^c	$0.98_{\pm 0.005}$	$0.98_{\pm 0.007}$	$0.97_{\pm 0.007}$	$0.95_{\pm 0.007}$	$0.96_{\pm 0.005}$	$0.98_{\pm 0.002}$	$0.97_{\pm 0.007}$	$0.97_{\pm 0.005}$	$0.98_{\pm 0.002}$	$0.98_{\pm 0.003}$	$0.97_{\pm 0.0}$

TABLE 3.23: Evasion Robustness of the ML-PWD against eWA^r and rWA^r on the `Zenodo`. The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the ML-PWD, while rows correspond to a specific eWsP or rWsP perturbation.

$\mathcal{A}$	F	no-atk	eWsP								rWsP			
			addImgBot	modFntTyp	modCpy	addIcn	delSusLnk	delSusFrm	modTtl	delCpy	modBgimg	modBgClr	modFntClr	modFntSiz
CN	F^u	$0.96_{\pm 0.007}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$
	F^r	$0.86_{\pm 0.013}$	$0.74_{\pm 0.06}$	$0.88_{\pm 0.013}$	$0.76_{\pm 0.045}$	$0.82_{\pm 0.019}$	$0.91_{\pm 0.016}$	$0.86_{\pm 0.036}$	$0.88_{\pm 0.013}$	$0.76_{\pm 0.045}$	$0.88_{\pm 0.013}$	$0.88_{\pm 0.013}$	$0.88_{\pm 0.013}$	$0.88_{\pm 0.013}$
	F^c	$0.97_{\pm 0.009}$	$0.97_{\pm 0.01}$	$0.97_{\pm 0.012}$	$0.97_{\pm 0.013}$	$0.96_{\pm 0.012}$	$0.96_{\pm 0.012}$	$0.97_{\pm 0.012}$	$0.97_{\pm 0.013}$	$0.97_{\pm 0.013}$	$0.97_{\pm 0.012}$	$0.97_{\pm 0.012}$	$0.97_{\pm 0.012}$	$0.97_{\pm 0.012}$
RF	F^u	$0.96_{\pm 0.007}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$
	F^r	$0.90_{\pm 0.013}$	$0.8_{\pm 0.045}$	$0.9_{\pm 0.013}$	$0.89_{\pm 0.014}$	$0.84_{\pm 0.025}$	$0.9_{\pm 0.015}$	$0.9_{\pm 0.014}$	$0.9_{\pm 0.013}$	$0.89_{\pm 0.014}$	$0.9_{\pm 0.013}$	$0.9_{\pm 0.013}$	$0.9_{\pm 0.013}$	$0.9_{\pm 0.013}$
	F^c	$0.97_{\pm 0.009}$	$0.98_{\pm 0.006}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.96_{\pm 0.006}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.004}$	$0.98_{\pm 0.004}$	$0.98_{\pm 0.004}$	$0.98_{\pm 0.005}$
LR	F^u	$0.97_{\pm 0.005}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$
	F^r	$0.80_{\pm 0.013}$	$0.9_{\pm 0.021}$	$0.8_{\pm 0.013}$	$0.8_{\pm 0.013}$	$0.88_{\pm 0.013}$	$0.74_{\pm 0.017}$	$0.8_{\pm 0.012}$	$0.79_{\pm 0.013}$	$0.8_{\pm 0.013}$	$0.8_{\pm 0.013}$	$0.8_{\pm 0.013}$	$0.8_{\pm 0.013}$	$0.8_{\pm 0.013}$
	F^c	$0.98_{\pm 0.005}$	$0.97_{\pm 0.008}$	$0.98_{\pm 0.001}$	$0.98_{\pm 0.002}$	$0.97_{\pm 0.008}$	$0.96_{\pm 0.002}$	$0.97_{\pm 0.008}$	$0.98_{\pm 0.002}$	$0.98_{\pm 0.001}$	$0.98_{\pm 0.001}$	$0.98_{\pm 0.001}$	$0.98_{\pm 0.002}$	$0.98_{\pm 0.005}$

TABLE 3.24: Impact of WA^u on ML-PWD of `Zenodo`. The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the ML-PWD, while rows correspond to a specific iWsP perturbation.

$\mathcal{A}$	F	no-atk	replChar	sepWrd	delChar	swpChar	addChar	atkPth
CN	F^u	$0.96_{\pm 0.007}$	$0.98_{\pm 0.012}$	$0.95_{\pm 0.024}$	$0.99_{\pm 0.009}$	$0.99_{\pm 0.013}$	$0.99_{\pm 0.007}$	$0.97_{\pm 0.017}$
	F^r	$0.86_{\pm 0.013}$	$0.5_{\pm 0.043}$	$0.49_{\pm 0.04}$	$0.5_{\pm 0.043}$	$0.49_{\pm 0.04}$	$0.49_{\pm 0.038}$	$0.5_{\pm 0.043}$
	F^c	$0.97_{\pm 0.009}$	$0.98_{\pm 0.017}$	$0.95_{\pm 0.025}$	$0.99_{\pm 0.024}$	$0.99_{\pm 0.019}$	$0.99_{\pm 0.017}$	$0.97_{\pm 0.021}$
RF	F^u	$0.96_{\pm 0.007}$	$1.0_{\pm 0.004}$	$0.98_{\pm 0.0}$	$1.0_{\pm 0.005}$	$1.0_{\pm 0.004}$	$1.0_{\pm 0.006}$	$0.99_{\pm 0.002}$
	F^r	$0.90_{\pm 0.013}$	$0.73_{\pm 0.043}$	$0.73_{\pm 0.043}$	$0.74_{\pm 0.043}$	$0.74_{\pm 0.041}$	$0.75_{\pm 0.041}$	$0.73_{\pm 0.043}$
	F^c	$0.97_{\pm 0.009}$	$1.0_{\pm 0.005}$	$0.99_{\pm 0.001}$	$1.0_{\pm 0.0}$	$1.0_{\pm 0.002}$	$1.0_{\pm 0.003}$	$0.98_{\pm 0.006}$
LR	F^u	$0.97_{\pm 0.005}$	$0.99_{\pm 0.002}$	$0.99_{\pm 0.003}$	$1.0_{\pm 0.003}$	$1.0_{\pm 0.0}$	$0.99_{\pm 0.001}$	$0.97_{\pm 0.007}$
	F^r	$0.80_{\pm 0.013}$	$0.78_{\pm 0.0}$	$0.79_{\pm 0.0}$	$0.79_{\pm 0.0}$	$0.79_{\pm 0.0}$	$0.8_{\pm 0.0}$	$0.78_{\pm 0.0}$
	F^c	$0.98_{\pm 0.005}$	$0.99_{\pm 0.0}$	$1.0_{\pm 0.001}$	$1.0_{\pm 0.0}$	$1.0_{\pm 0.0}$	$1.0_{\pm 0.0}$	$0.97_{\pm 0.004}$

TABLE 3.25: Impact of PA^r and MA^r on ML-PWD generated on `Zenodo`. The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the ML-PWD, while rows correspond to a specific PsP or MsP attack.

$\mathcal{A}$	F	no-atk	PsP							MsP					
			delTxt	delFrm	delSpn	delTtl	addLngTxt	delFtr	replSusFtrLnk	brTg	delHt	delHd	delBdy	brTgs	hmg
CN	F^u	$0.96_{\pm 0.007}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.97_{\pm 0.014}$
	F^r	$0.86_{\pm 0.013}$	$0.78_{\pm 0.046}$	$0.86_{\pm 0.036}$	$0.88_{\pm 0.015}$	$0.88_{\pm 0.013}$	$0.88_{\pm 0.013}$	$0.64_{\pm 0.052}$	$0.74_{\pm 0.05}$	$0.88_{\pm 0.013}$	$0.0_{\pm 0.0}$	$0.82_{\pm 0.011}$	$0.43_{\pm 0.049}$	$0.88_{\pm 0.013}$	$0.4_{\pm 0.035}$
	F^c	$0.97_{\pm 0.009}$	$0.97_{\pm 0.012}$	$0.97_{\pm 0.012}$	$0.97_{\pm 0.012}$	$0.98_{\pm 0.011}$	$0.97_{\pm 0.012}$	$0.96_{\pm 0.017}$	$0.96_{\pm 0.012}$	$0.97_{\pm 0.012}$	$0.9_{\pm 0.033}$	$0.95_{\pm 0.02}$	$0.93_{\pm 0.019}$	$0.97_{\pm 0.012}$	$0.92_{\pm 0.026}$
RF	F^u	$0.96_{\pm 0.007}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.97_{\pm 0.007}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.006}$
	F^r	$0.90_{\pm 0.013}$	$0.86_{\pm 0.032}$	$0.9_{\pm 0.014}$	$0.9_{\pm 0.012}$	$0.86_{\pm 0.032}$	$0.9_{\pm 0.013}$	$0.86_{\pm 0.018}$	$0.84_{\pm 0.012}$	$0.9_{\pm 0.013}$	$0.0_{\pm 0.0}$	$0.66_{\pm 0.083}$	$0.46_{\pm 0.024}$	$0.9_{\pm 0.013}$	$0.22_{\pm 0.066}$
	F^c	$0.97_{\pm 0.009}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.004}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.004}$	$0.97_{\pm 0.006}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.006}$	$0.96_{\pm 0.006}$	$0.98_{\pm 0.005}$	$0.96_{\pm 0.006}$	$0.98_{\pm 0.005}$	$0.97_{\pm 0.007}$
LR	F^u	$0.97_{\pm 0.005}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.009}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.009}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.98_{\pm 0.005}$
	F^r	$0.80_{\pm 0.013}$	$0.66_{\pm 0.004}$	$0.8_{\pm 0.012}$	$0.8_{\pm 0.013}$	$0.66_{\pm 0.004}$	$0.8_{\pm 0.013}$	$0.76_{\pm 0.008}$	$0.73_{\pm 0.022}$	$0.8_{\pm 0.013}$	$0.0_{\pm 0.0}$	$0.74_{\pm 0.0}$	$0.32_{\pm 0.006}$	$0.8_{\pm 0.013}$	$0.24_{\pm 0.02}$
	F^c	$0.98_{\pm 0.005}$	$0.96_{\pm 0.005}$	$0.97_{\pm 0.008}$	$0.98_{\pm 0.002}$	$0.96_{\pm 0.007}$	$0.98_{\pm 0.002}$	$0.98_{\pm 0.001}$	$0.97_{\pm 0.006}$	$0.97_{\pm 0.005}$	$0.96_{\pm 0.006}$	$0.97_{\pm 0.005}$	$0.97_{\pm 0.003}$	$0.98_{\pm 0.001}$	$0.96_{\pm 0.005}$

TABLE 3.26: Impact of PA^r+PA^r on Zenodo

$\mathcal{A}$	F	no-atk	addLngTxt_delTtl	delFtr_delTtl	delFtr_addLngTxt	delSpn_delTtl	delSpn_delFtrer	delSpn_addLngTxt	delFrm_delFtr	delFrm_delSpn
CN	F^u	$0.96_{\pm 0.007}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$
	F^r	$0.86_{\pm 0.013}$	$0.88_{\pm 0.013}$	$0.64_{\pm 0.052}$	$0.64_{\pm 0.052}$	$0.88_{\pm 0.015}$	$0.64_{\pm 0.052}$	$0.88_{\pm 0.015}$	$0.78_{\pm 0.064}$	$0.83_{\pm 0.045}$
	F^c	$0.97_{\pm 0.009}$	$0.98_{\pm 0.011}$	$0.96_{\pm 0.015}$	$0.96_{\pm 0.017}$	$0.98_{\pm 0.011}$	$0.96_{\pm 0.016}$	$0.98_{\pm 0.012}$	$0.96_{\pm 0.015}$	$0.97_{\pm 0.015}$
RF	F^u	$0.96_{\pm 0.007}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$
	F^r	$0.90_{\pm 0.013}$	$0.86_{\pm 0.032}$	$0.8_{\pm 0.042}$	$0.86_{\pm 0.018}$	$0.87_{\pm 0.025}$	$0.86_{\pm 0.018}$	$0.9_{\pm 0.012}$	$0.7_{\pm 0.052}$	$0.79_{\pm 0.034}$
	F^c	$0.97_{\pm 0.009}$	$0.98_{\pm 0.004}$	$0.97_{\pm 0.007}$	$0.97_{\pm 0.006}$	$0.98_{\pm 0.005}$	$0.97_{\pm 0.007}$	$0.98_{\pm 0.005}$	$0.97_{\pm 0.007}$	$0.98_{\pm 0.005}$
LR	F^u	$0.97_{\pm 0.005}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$
	F^r	$0.80_{\pm 0.013}$	$0.66_{\pm 0.004}$	$0.58_{\pm 0.005}$	$0.76_{\pm 0.008}$	$0.65_{\pm 0.004}$	$0.76_{\pm 0.008}$	$0.8_{\pm 0.013}$	$0.7_{\pm 0.009}$	$0.76_{\pm 0.005}$
	F^c	$0.98_{\pm 0.005}$	$0.96_{\pm 0.005}$	$0.97_{\pm 0.01}$	$0.98_{\pm 0.004}$	$0.96_{\pm 0.007}$	$0.97_{\pm 0.005}$	$0.97_{\pm 0.008}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$

TABLE 3.27: Impact of PA^r+WA^r on ML-PWD generated on Zenodo. The cells report the average (and std. dev.) tpr over the 50 reiterations. Lines correspond to the ML-PWD, while rows correspond to a specific PsP+WsP perturbation.

(A) PA^r+WA^r

$\mathcal{A}$	F	no-atk	addLngTxt_ addInLnk	addLngTxt_ delHidIt	addLngTxt_ replOnfoc	addLngTxt_ replPass	addLngTxt_ addHidP	addLngTxt_ replJS	addLngTxt_ delSusLnk	addLngTxt_ modCpy	addLngTxt_ modTtl	addLngTxt_ addIcn	addLngTxt_ addSusLnk	addLngTxt_ addImgBot	addLngTxt_ modBgClr
CN	F^u	$0.96_{\pm 0.007}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$
	F^r	$0.86_{\pm 0.013}$	$0.61_{\pm 0.054}$	$0.88_{\pm 0.012}$	$0.88_{\pm 0.013}$	$0.88_{\pm 0.013}$	$0.28_{\pm 0.008}$	$0.74_{\pm 0.05}$	$0.91_{\pm 0.016}$	$0.76_{\pm 0.043}$	$0.88_{\pm 0.013}$	$0.82_{\pm 0.019}$	$0.5_{\pm 0.043}$	$0.72_{\pm 0.058}$	$0.88_{\pm 0.013}$
	F^c	$0.97_{\pm 0.009}$	$0.97_{\pm 0.021}$	$0.97_{\pm 0.016}$	$0.98_{\pm 0.013}$	$0.98_{\pm 0.013}$	$0.92_{\pm 0.024}$	$0.95_{\pm 0.015}$	$0.96_{\pm 0.012}$	$0.98_{\pm 0.013}$	$0.98_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.97_{\pm 0.02}$	$0.96_{\pm 0.013}$	$0.98_{\pm 0.013}$
RF	F^u	$0.96_{\pm 0.007}$	$0.97_{\pm 0.007}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$
	F^r	$0.90_{\pm 0.013}$	$0.95_{\pm 0.018}$	$0.84_{\pm 0.01}$	$0.9_{\pm 0.013}$	$0.9_{\pm 0.013}$	$0.03_{\pm 0.064}$	$0.71_{\pm 0.016}$	$0.9_{\pm 0.015}$	$0.89_{\pm 0.014}$	$0.9_{\pm 0.013}$	$0.84_{\pm 0.025}$	$0.73_{\pm 0.043}$	$0.78_{\pm 0.044}$	$0.9_{\pm 0.013}$
	F^c	$0.97_{\pm 0.009}$	$0.98_{\pm 0.008}$	$0.98_{\pm 0.004}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.004}$	$0.96_{\pm 0.006}$	$0.96_{\pm 0.007}$	$0.96_{\pm 0.007}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.004}$	$0.98_{\pm 0.004}$	$0.97_{\pm 0.006}$	$0.98_{\pm 0.006}$	$0.98_{\pm 0.004}$
LR	F^u	$0.97_{\pm 0.005}$	$0.97_{\pm 0.009}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$
	F^r	$0.80_{\pm 0.013}$	$0.96_{\pm 0.005}$	$0.79_{\pm 0.014}$	$0.8_{\pm 0.013}$	$0.8_{\pm 0.013}$	$0.24_{\pm 0.019}$	$0.46_{\pm 0.013}$	$0.74_{\pm 0.017}$	$0.8_{\pm 0.013}$	$0.79_{\pm 0.013}$	$0.88_{\pm 0.013}$	$0.78_{\pm 0.0}$	$0.88_{\pm 0.021}$	$0.8_{\pm 0.013}$
	F^c	$0.98_{\pm 0.005}$	$0.99_{\pm 0.004}$	$0.98_{\pm 0.002}$	$0.98_{\pm 0.002}$	$0.98_{\pm 0.001}$	$0.96_{\pm 0.008}$	$0.95_{\pm 0.007}$	$0.95_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.001}$	$0.97_{\pm 0.0}$	$0.97_{\pm 0.005}$	$0.98_{\pm 0.005}$

(B) PA^r+WA^r

$\mathcal{A}$	F	no-atk	delFtr_ addInLnk	delFtr_ delHidIt	delFtr_ replOnfoc	delFtr_ replPass	delFtr_ addHidP	delFtr_ replJS	delFtr_ delSusLnk	delFtr_ modCpy	delFtr_ modTtl	delFtr_ addIcn	delFtr_ addSusLnk	delFtr_ modBgClr
CN	F^u	$0.96_{\pm 0.007}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.94_{\pm 0.03}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$
	F^r	$0.86_{\pm 0.013}$	$0.33_{\pm 0.053}$	$0.64_{\pm 0.052}$	$0.64_{\pm 0.052}$	$0.54_{\pm 0.049}$	$0.54_{\pm 0.049}$	$0.64_{\pm 0.052}$	$0.85_{\pm 0.057}$	$0.63_{\pm 0.05}$	$0.64_{\pm 0.052}$	$0.55_{\pm 0.055}$	$0.28_{\pm 0.02}$	$0.64_{\pm 0.052}$
	F^c	$0.97_{\pm 0.009}$	$0.96_{\pm 0.028}$	$0.96_{\pm 0.016}$	$0.96_{\pm 0.017}$	$0.83_{\pm 0.035}$	$0.91_{\pm 0.025}$	$0.94_{\pm 0.012}$	$0.96_{\pm 0.011}$	$0.96_{\pm 0.017}$	$0.96_{\pm 0.017}$	$0.95_{\pm 0.015}$	$0.96_{\pm 0.027}$	$0.96_{\pm 0.017}$
RF	F^u	$0.96_{\pm 0.007}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.96_{\pm 0.007}$	$0.96_{\pm 0.007}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.003}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$
	F^r	$0.90_{\pm 0.013}$	$0.95_{\pm 0.016}$	$0.8_{\pm 0.024}$	$0.86_{\pm 0.018}$	$0.15_{\pm 0.075}$	$0.15_{\pm 0.075}$	$0.63_{\pm 0.054}$	$0.82_{\pm 0.024}$	$0.86_{\pm 0.019}$	$0.86_{\pm 0.018}$	$0.82_{\pm 0.031}$	$0.59_{\pm 0.07}$	$0.86_{\pm 0.018}$
	F^c	$0.97_{\pm 0.009}$	$0.98_{\pm 0.008}$	$0.97_{\pm 0.006}$	$0.97_{\pm 0.007}$	$0.94_{\pm 0.018}$	$0.94_{\pm 0.018}$	$0.96_{\pm 0.006}$	$0.96_{\pm 0.007}$	$0.98_{\pm 0.006}$	$0.97_{\pm 0.006}$	$0.97_{\pm 0.007}$	$0.96_{\pm 0.006}$	$0.97_{\pm 0.006}$
LR	F^u	$0.97_{\pm 0.005}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.96_{\pm 0.0}$	$0.96_{\pm 0.0}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.009}$	$0.97_{\pm 0.004}$
	F^r	$0.80_{\pm 0.013}$	$0.97_{\pm 0.0}$	$0.75_{\pm 0.006}$	$0.76_{\pm 0.008}$	$0.37_{\pm 0.01}$	$0.37_{\pm 0.01}$	$0.45_{\pm 0.008}$	$0.74_{\pm 0.011}$	$0.76_{\pm 0.008}$	$0.75_{\pm 0.008}$	$0.86_{\pm 0.007}$	$0.72_{\pm 0.0}$	$0.76_{\pm 0.008}$
	F^c	$0.98_{\pm 0.005}$	$0.98_{\pm 0.007}$	$0.97_{\pm 0.007}$	$0.97_{\pm 0.008}$	$0.96_{\pm 0.007}$	$0.96_{\pm 0.007}$	$0.96_{\pm 0.004}$	$0.95_{\pm 0.005}$	$0.98_{\pm 0.004}$	$0.98_{\pm 0.002}$	$0.98_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.008}$

(C) PA^r+WA^r

$\mathcal{A}$	F	no-atk	delFrm_ addInLnk	delFrm_ onclick	delFrm_ delHidIt	delFrm_ addHidP	delFrm_ replJS	delFrm_ delSusLnk	delFrm_ addImgBot	delFrm_ modFntSiz	delFrm_ modBgimg	delFrm_ modBgClr	delFrm_ delCpy	delFrm_ modTtl	delFrm_ modCpy	delFrm_ addIcn	delFrm_ replRet
CN	F^u	$0.96_{\pm 0.007}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.94_{\pm 0.018}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$
	F^r	$0.86_{\pm 0.013}$	$0.56_{\pm 0.043}$	$0.87_{\pm 0.024}$	$0.84_{\pm 0.038}$	$0.27_{\pm 0.008}$	$0.76_{\pm 0.054}$	$0.87_{\pm 0.028}$	$0.75_{\pm 0.052}$	$0.84_{\pm 0.038}$	$0.84_{\pm 0.038}$	$0.84_{\pm 0.038}$	$0.78_{\pm 0.064}$	$0.84_{\pm 0.038}$	$0.78_{\pm 0.064}$	$0.82_{\pm 0.042}$	$0.84_{\pm 0.038}$
	F^c	$0.97_{\pm 0.009}$	$0.96_{\pm 0.025}$	$0.96_{\pm 0.008}$	$0.97_{\pm 0.015}$	$0.92_{\pm 0.022}$	$0.91_{\pm 0.027}$	$0.96_{\pm 0.011}$	$0.96_{\pm 0.018}$	$0.97_{\pm 0.016}$	$0.97_{\pm 0.016}$	$0.97_{\pm 0.016}$	$0.97_{\pm 0.016}$	$0.97_{\pm 0.016}$	$0.97_{\pm 0.016}$	$0.96_{\pm 0.013}$	$0.97_{\pm 0.016}$
RF	F^u	$0.96_{\pm 0.007}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.003}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.003}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$
	F^r	$0.90_{\pm 0.013}$	$0.9_{\pm 0.043}$	$0.8_{\pm 0.031}$	$0.81_{\pm 0.028}$	$0.01_{\pm 0.045}$	$0.58_{\pm 0.05}$	$0.79_{\pm 0.027}$	$0.69_{\pm 0.065}$	$0.82_{\pm 0.023}$	$0.82_{\pm 0.023}$	$0.82_{\pm 0.023}$	$0.72_{\pm 0.041}$	$0.82_{\pm 0.023}$	$0.72_{\pm 0.041}$	$0.77_{\pm 0.035}$	$0.82_{\pm 0.023}$
	F^c	$0.97_{\pm 0.009}$	$0.98_{\pm 0.007}$	$0.97_{\pm 0.009}$	$0.98_{\pm 0.005}$	$0.96_{\pm 0.005}$	$0.96_{\pm 0.006}$	$0.96_{\pm 0.007}$	$0.97_{\pm 0.006}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.99_{\pm 0.006}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$
LR	F^u	$0.97_{\pm 0.005}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.009}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$
	F^r	$0.80_{\pm 0.013}$	$0.93_{\pm 0.009}$	$0.6_{\pm 0.024}$	$0.76_{\pm 0.002}$	$0.24_{\pm 0.019}$	$0.36_{\pm 0.005}$	$0.69_{\pm 0.011}$	$0.87_{\pm 0.027}$	$0.77_{\pm 0.005}$	$0.77_{\pm 0.005}$	$0.77_{\pm 0.005}$	$0.77_{\pm 0.005}$	$0.76_{\pm 0.005}$	$0.77_{\pm 0.005}$	$0.88_{\pm 0.008}$	$0.77_{\pm 0.005}$
	F^c	$0.98_{\pm 0.005}$	$0.98_{\pm 0.0}$	$0.97_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.97_{\pm 0.008}$	$0.96_{\pm 0.007}$	$0.95_{\pm 0.005}$	$0.97_{\pm 0.005}$	$0.97_{\pm 0.008}$	$0.98_{\pm 0.002}$	$0.98_{\pm 0.001}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.001}$	$0.97_{\pm 0.005}$	$0.98_{\pm 0.005}$

(D) PA^r+WA^r

$\mathcal{A}$	F	no-atk	delSpn_ addInLnk	delSpn_ replOnfoc	delSpn_ replPass	delSpn_ addHidP	delSpn_ replJS	delSpn_ delSusLnk	delSpn_ modCpy	delSpn_ modTtl	delSpn_ addIcn	delSpn_ addSusLnk	delSpn_ addImgBot	delSpn_ modBgClr
CN	F^u	$0.96_{\pm 0.007}$	$0.94_{\pm 0.021}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$
	F^r	$0.86_{\pm 0.013}$	$0.63_{\pm 0.025}$	$0.88_{\pm 0.015}$	$0.88_{\pm 0.014}$	$0.27_{\pm 0.008}$	$0.74_{\pm 0.052}$	$0.89_{\pm 0.019}$	$0.76_{\pm 0.044}$	$0.88_{\pm 0.015}$	$0.83_{\pm 0.02}$	$0.4_{\pm 0.02}$	$0.75_{\pm 0.062}$	$0.88_{\pm 0.015}$
	F^c	$0.97_{\pm 0.009}$	$0.95_{\pm 0.02}$	$0.98_{\pm 0.012}$	$0.98_{\pm 0.011}$	$0.92_{\pm 0.024}$	$0.95_{\pm 0.015}$	$0.96_{\pm 0.012}$	$0.98_{\pm 0.012}$	$0.98_{\pm 0.012}$	$0.96_{\pm 0.013}$	$0.95_{\pm 0.019}$	$0.97_{\pm 0.011}$	$0.98_{\pm 0.012}$
RF	F^u	$0.96_{\pm 0.007}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$
	F^r	$0.90_{\pm 0.013}$	$0.95_{\pm 0.021}$	$0.9_{\pm 0.012}$	$0.9_{\pm 0.012}$	$0.03_{\pm 0.068}$	$0.66_{\pm 0.016}$	$0.89_{\pm 0.016}$	$0.9_{\pm 0.013}$	$0.9_{\pm 0.012}$	$0.85_{\pm 0.027}$	$0.58_{\pm 0.043}$	$0.8_{\pm 0.051}$	$0.9_{\pm 0.012}$
	F^c	$0.97_{\pm 0.009}$	$0.98_{\pm 0.007}$	$0.98_{\pm 0.004}$	$0.98_{\pm 0.004}$	$0.96_{\pm 0.006}$	$0.96_{\pm 0.006}$	$0.96_{\pm 0.006}$	$0.98_{\pm 0.004}$	$0.98_{\pm 0.004}$	$0.98_{\pm 0.004}$	$0.97_{\pm 0.007}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.004}$
LR	F^u	$0.97_{\pm 0.005}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$
	F^r	$0.80_{\pm 0.013}$	$0.96_{\pm 0.005}$	$0.8_{\pm 0.013}$	$0.8_{\pm 0.013}$	$0.23_{\pm 0.017}$	$0.41_{\pm 0.013}$	$0.74_{\pm 0.017}$	$0.8_{\pm 0.013}$	$0.79_{\pm 0.013}$	$0.88_{\pm 0.013}$	$0.62_{\pm 0.0}$	$0.9_{\pm 0.021}$	$0.8_{\pm 0.013}$
	F^c	$0.98_{\pm 0.005}$	$0.98_{\pm 0.003}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.97_{\pm 0.009}$	$0.96_{\pm 0.004}$	$0.96_{\pm 0.002}$	$0.98_{\pm 0.0}$	$0.98_{\pm 0.001}$	$0.98_{\pm 0.001}$	$0.97_{\pm 0.0}$	$0.98_{\pm 0.003}$	$0.98_{\pm 0.0}$

TABLE 3.28: Impact of PA^r+MA^r in Zenodo

$\mathcal{A}$	F	no-atk	addLngTxt_delBdy	delFtr_delBdy	delSpn_delBdy
CN	F^u	$0.96_{\pm 0.007}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$
	F^r	$0.86_{\pm 0.013}$	$0.88_{\pm 0.013}$	$0.64_{\pm 0.052}$	$0.88_{\pm 0.015}$
	F^c	$0.97_{\pm 0.009}$	$0.98_{\pm 0.013}$	$0.96_{\pm 0.017}$	$0.98_{\pm 0.012}$
RF	F^u	$0.96_{\pm 0.007}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$
	F^r	$0.90_{\pm 0.013}$	$0.9_{\pm 0.013}$	$0.86_{\pm 0.018}$	$0.9_{\pm 0.012}$
	F^c	$0.97_{\pm 0.009}$	$0.98_{\pm 0.005}$	$0.97_{\pm 0.006}$	$0.98_{\pm 0.005}$
LR	F^u	$0.97_{\pm 0.005}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$
	F^r	$0.80_{\pm 0.013}$	$0.8_{\pm 0.013}$	$0.76_{\pm 0.008}$	$0.8_{\pm 0.013}$
	F^c	$0.98_{\pm 0.005}$	$0.98_{\pm 0.002}$	$0.98_{\pm 0.004}$	$0.98_{\pm 0.001}$

TABLE 3.29: Impact of WA^r+WA^r on Zenodo

$\mathcal{A}$	F	no-atk	replOnfoc_replRet	htEsc_replRet	htEncd_replRet
CN	F^u	$0.96_{\pm 0.007}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$	$0.96_{\pm 0.013}$
	F^r	$0.86_{\pm 0.013}$	$0.88_{\pm 0.013}$	$0.87_{\pm 0.025}$	$0.0_{\pm 0.0}$
	F^c	$0.97_{\pm 0.009}$	$0.98_{\pm 0.013}$	$0.96_{\pm 0.007}$	$0.91_{\pm 0.032}$
RF	F^u	$0.96_{\pm 0.007}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$	$0.98_{\pm 0.005}$
	F^r	$0.90_{\pm 0.013}$	$0.9_{\pm 0.013}$	$0.84_{\pm 0.027}$	$0.0_{\pm 0.0}$
	F^c	$0.97_{\pm 0.009}$	$0.98_{\pm 0.005}$	$0.96_{\pm 0.007}$	$0.96_{\pm 0.007}$
LR	F^u	$0.97_{\pm 0.005}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$	$0.97_{\pm 0.004}$
	F^r	$0.80_{\pm 0.013}$	$0.8_{\pm 0.013}$	$0.55_{\pm 0.009}$	$0.0_{\pm 0.0}$
	F^c	$0.98_{\pm 0.005}$	$0.97_{\pm 0.005}$	$0.97_{\pm 0.001}$	$0.97_{\pm 0.004}$

Part II

User perceptions on Phishing webpages

Chapter 4

Understanding User Perceptions of Adversarial Phishing Websites

After nearly three decades of research [112], phishing attacks are still rampant. According to the FBI's 2022 crime data [21], phishing is the topmost form of cybercrime, with reported victim loss allegedly increasing by over 1000% since 2018. In this context, phishing *websites* are a type of online scam used by attackers to steal sensitive information such as login credentials, financial information, or personal data. To increase their effectiveness, phishing websites aim to mimic legitimate ones [44], thereby tricking unaware and distracted victims—who may not notice subtle differences in their appearance.

Recently, numerous automatic Phishing Website Detectors (PWD) have been proposed, which can rely on blocklists [199], or be entirely data-driven [60]. The former works by checking whether a given website is included in their (public or private) blocklist, which consists of URLs (collected, e.g., from well-known repositories—such as PhishTank [26]). However, blocklist-based anti-phishing methods, despite their low false positive rates, cannot detect "novel" phishing websites [247]. These shortcomings can be compensated via data-driven techniques. Among these, Machine Learning (ML) algorithms seek to autonomously learn (by "training" on a given dataset) to identify patterns that may not be discernible to the human eye. The remarkable performance of ML methods in computer vision [163] led to many efforts to investigate their effectiveness in various fields—including that of phishing website detection. In particular, ML-based phishing website detectors (ML-PWD) can detect previously unseen phishing websites while maintaining low rates of false positives [60], which can be achieved by analyzing either textual or visual features from any given webpage (e.g., [77, 179]).

Motivation. Machine learning has now become mainstream even for the detection of phishing webpages [107]. However, ML is prone to evasion attacks, which entail crafting an "adversarial phishing website" (APW) by introducing imperceptible perturbations (located, e.g., in the HTML [60], or in some visual element [164] of a webpage) that fool an ML-PWD. Unfortunately, security practitioners persist in not addressing such a threat [54] (despite abundant alarms from academia [206, 221]). In this context, we observe that recent interview studies [79, 126, 185] about adversarial ML (AML) in practice are based on the participants' (self-reported) understanding of AML's concepts, thereby focusing on the question "What is the practitioners' awareness of AML?". We argue that to (i) establish whether AML is truly a threat and, if so, (ii) convince practitioners to take AML into consideration while designing their ML systems, the focus should be on the question "*What is the impact of AML on the end-users in practice? That is: does AML fool users as much as it fools ML models?*".

This paper revolves around investigating this dilemma for phishing website detection. Compared to existing works that only focus on using AML to attack ML-PWD (e.g., [60, 164]), our work advances existing knowledge by examining how *human users* perceive adversarial phishing webpages that evade ML-PWD.

Problem Statement. To explore the users' perception of APW, our paper revolves around answering four research questions (RQ):

RQ1 Are adversarially perturbed phishing webpages more easily detectable by users—w.r.t. unperturbed ones? (§4.4.2)

RQ2 Are some perturbations more likely to deceive users? (§4.4.2)

RQ3 How much do users' background (e.g., age, gender, expertise) correlates with their phishing detection skills? (§4.4.3)

RQ4 What cues do users typically look for (and potentially rely on) to judge the legitimacy of any given website? (§4.5)

To answer our RQ, we conduct (§4.3) two user studies (n=470). The first focuses on assessing how well users can distinguish legitimate webpages from "unperturbed" phishing webpages. The second is to assess how well users can distinguish "adversarial" phishing webpages from legitimate webpages. Overall, we obtained over 7k responses encompassing various classes of webpages including: legitimate and 'unperturbed' phishing webpages, four types of APW (crafted through well-known AML techniques), as well as APW "from the wild Web" that bypassed production-grade ML-PWD (§4.2).

Contributions. After analysing the results of our user studies both quantitatively and qualitatively, we derive three key-findings.

1. **Adversarial phishing is a threat to both users and ML.** In particular, three out of the four adversarial perturbations we considered have comparable effectiveness in deceiving users when compared to unperturbed phishing webpages—but the latter cannot bypass the ML-PWD. We argue that user studies are a necessary step that is currently missing in most AML research on phishing detectors (see §4.1). Specifically, *it is crucial to compare adversarial phishing webpages with unperturbed phishing webpages* to make sure APW do not sacrifice effectiveness against users in favor of an improved evasion rate.

2. **Not all adversarial perturbations are equally effective.** In particular, adversarial webpages with added typos are more noticeable to users, as confirmed by statistical tests. The reasoning provided by participants also indicates that textual indicators play a major role in their decision-making process. In addition, we verify that adversarial phishing pages "from the wild Web" (which bypassed production-grade ML-PWD) are *more detectable by users than unperturbed phishing pages*.

3. As a surprising and counter-intuitive observation, **users' self-reported frequency of visiting a brand's website has a statistically significant *negative* correlation with their phishing detection accuracy**. Users who claimed to frequently visit websites of a given brand performed worse on the phishing webpages targeting this brand. We suspect this is correlated to prior findings that familiarity leads to overconfidence [210, 256]

Finally, our work can serve as a benchmark for future research on evasion attacks against ML-PWD, since it facilitates *assessing the effectiveness on end users* of the proposed attacks. To this purpose, we release our user study questionnaires, codebook, data, and code we developed [3]. We will also submit our tools for artifact evaluation.

4.1 Background and Related Work

To set the stage for our contribution, we raise the attention on some simple security concepts, which we use as a scaffold to position our paper within existing literature. We provide exhaustive background (covering ML-PWD and adversarial ML) in Appendix 4.10.

Phishing in a Nutshell. From a security standpoint, the goal of a phisher (i.e., the attacker) is to trick a *human user* to, e.g., input their private (or sensitive) data, or click on a malicious link.

> **REMARK:** bypassing a given detector (despite being necessary) is not sufficient for a phishing webpage to be successful.

Given the above, all those papers (e.g., [60, 64, 98, 164, 190]) showing that ML-PWD can be evaded via "adversarial perturbations" – while useful for investigating some robustness properties of ML – could hardly provide a compelling case that "adversarial examples are a problem *in reality*". Indeed, doing so would necessitate a double form of assessment, entailing both machine and human: first, it is necessary to craft an adversarial webpage and show that it bypasses a functional ML-PWD (i.e., a false negative); then, it is necessary to assess whether humans (i.e., the true target of phishing) are still tricked by such a webpage. Perhaps surprisingly, however, *such systematic assessments are missing from current literature*.

Research Gap. Scientific literature on phishing defense can be divided in two categories: *technical papers* (e.g., [60, 164, 171, 175, 179]), which propose (or attack) a given solution; and *user studies* (e.g., [53, 123, 263]), which seek to investigate the response of humans to phishing (useful for phishing training and education). However, to the best of our knowledge, none of these categories have questioned how humans respond to phishing webpages crafted to bypass ML-PWDs. Indeed, from an "adversarial ML" perspective, technical papers typically stop after showing that a given ML-PWD has been evaded (e.g., [64, 190]); whereas user studies either entailed "phishing" webpages that have been crafted ad-hoc (e.g., [123, 192]) or, even when real phishing webpages were considered (e.g., [53, 66]), the role of ML was irrelevant. Hence, the question: *"Are adversarial webpages a problem in reality?"* is still open. As a matter of fact, recent findings [54] revealed that the ML-PWD of a security company had over 9k false negatives in one month—some of which entailed "perturbations" that most laymen would notice (see Fig. 4.5).

Related Work. We acknowledge, however, that the limitations of prior work are well-justified. Indeed, technical papers can be complex, and carrying out user studies *on top* of devising a scientifically sound and relevant contribution is challenging; whereas user studies require the availability of ML-powered PWD, which are becoming popular only in recent years. Nonetheless, we found *two works which partially overlap* with ours. **(1)** Abdelnabi et al. [44], after proposing an ML-PWD, discuss a user study (in the Appendix, with limited details) wherein participants were shown

the webpages that bypassed the proposed ML-PWD and asked to rate "how trustworthy" such webpages were. The purpose of the user study, however, is to assess user agreements with their proposed similarity metric, and thus it does not involve the assessment of adversarial phishing pages or their comparison with benign/unperturbed phishing pages. **(2)** Lee et al. [164] attack an ML-PWD which exclusively focuses on the logo of well-known brands, and then carry out a user study asking participants how similar an adversarial logo was w.r.t. an original logo: the problem is that the logo is only a single element in a webpage (i.e., the webpage could be still detected by other automated mechanisms).

Our Goal. In this paper, we seek to overcome the shortcomings of prior work. Specifically, we investigate the response of human users to "adversarial" phishing webpages[1] that evaded ML-PWD (both real ones and custom-made); then, we compare such results with the ones from user assessments of "non-adversarial" phishing webpages. The rationale is that attackers are less interested in crafting adversarial webpages that, despite evading ML-PWD, can be easily spotted by end-users—i.e., their final target.

4.2 Data Collection & Generation

To answer our research questions, we design user studies wherein participants are asked to examine a mixed set of phishing and legitimate webpages. A crucial part of our research is that we want to investigate the response of users to adversarial webpages that *bypassed* ML-based detectors (both synthetic ones, as well as real products); indeed, this is necessary to determine whether adversarial webpages represent a problem "in reality". Therefore, before describing our user studies, we explain how we obtained a set of adversarial webpages that we can use for our user studies. Fig. 4.1 summarizes the workflow of our experimental methodology.

Overview. We seek to identify adversarial webpages that bypass either production-grade ML-PWD, or state-of-the-art research proposals. To meet this twofold requirement, we must first obtain a dataset including both benign and phishing webpages—which will be used to develop a custom ML-PWD. Then, after ensuring that our ML-PWD obtains good detection performance (i.e., high true positive rate with low false positive rate) in "non-adversarial" scenarios, we will use the phishing webpages in our dataset as the basis to craft adversarial phishing webpages. Such adversarial examples will then be tested against our custom ML-PWD. If they can evade the detection, we will consider them for our user study.

Dataset. To develop a state-of-the-art ML-PWD, we rely on the phishing dataset by Chiew et al. [94]. This dataset (used also, e.g., in [223]) contains 30k webpages: 15k are benign (source: Alexa top) and 15k are phishing (source: Phishtank [26]). We consider this dataset because, for each sample, it provides the HTML content as well as supporting files (e.g., CSS) and all the image components. This allows us to craft *realizable* perturbations on these webpages, thereby yielding adversarial webpages with high realistic fidelity. Other existing datasets (e.g., [64, 179]) do not allow this, since they lack CSS and/or image files. Finally, although our chosen dataset was released in 2018, its webpages still resemble the ones of the "current" version (as of Sept. 2023) of the corresponding websites.

[1] We **focus on phishing "on the Web"**. Other forms of phishing (such as via email [228] or phone calls [69]) and their detection (with or without ML) are orthogonal research areas to this paper (albeit some of our findings can be relevant also to these areas).

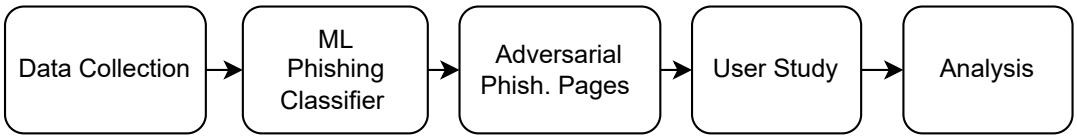
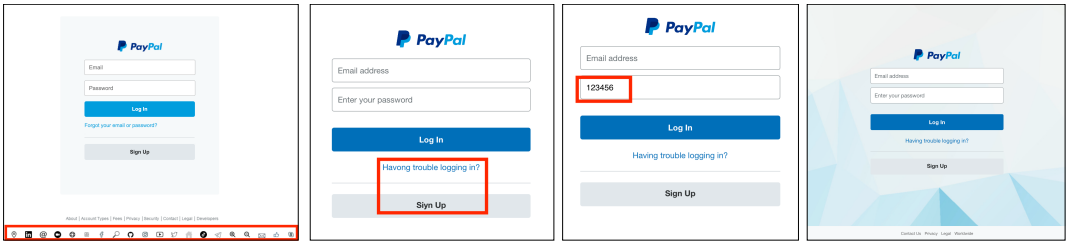


FIGURE 4.1: Workflow of our study.



(a) APW-Lab_img (b) APW-Lab_typo (c) APW-Lab_pswd (d) APW-Lab_bg

FIGURE 4.2: Example screenshot of lab-generated adversarial phishing pages targeting Paypal. We include two types of perturbations: (a) adding small images to the footer, (b) introducing typos, (c) making the password visible, and (d) adding a background image.

Custom ML-PWD. We first use the dataset [94] of benign and phishing webpages to train a ML-PWD. Then we add perturbations to a phishing webpage, aiming to trigger a false negative by the ML-PWD. In more detail, our ML-PWD relies on the random forest algorithm (thanks to its superior performance over other ML algorithms, as reported by many prior works [60, 247]). In particular, we rely on the code (and features) provided by [60] to develop our ML-PWD, for which we use 80% of the dataset for training and use the remaining 20% for testing. Our ML-PWD obtains performance comparable with the state-of-the-art, having a true positive rate of 0.98 and a false positive rate of 0.04 (results aligning with prior works [60, 223]). These results confirm that our ML-PWD (which we release [3]) *is a valid candidate for our research.*

Custom Adversarial Phishing Webpages. We use/adapt existing AML methods (borrowed from [60] and [269]) to generate 4 types of adversarial phishing webpages "in a lab" (*APW-Lab*). More specifically, we selected *four* types of perturbations that help a phishing page evade our custom ML-PWD, each yielding an adversarial phishing webpage having diverse visual cues:

1. *APW-Lab_img*: we insert a small array of images to the bottom of the web page (footer), as shown in Fig. 4.2(a).
2. *APW-Lab_typo*: we randomly insert typos to the text content of the web page as shown in Fig. 4.2(b).
3. *APW-Lab_pswd*: we make the password visible for the password input box, as shown in Fig. 4.2(c).
4. *APW-Lab_bg*: we randomly add a background image to the web page, as shown in Fig. 4.2(d).

The *APW-Lab* that bypass our ML-PWD will be used for the user study. We note that related work from Lee et al. [164] did not evaluate webpages but focused on logos only.

Real Adversarial Phishing Webpages. A prior work [54] identified 100 adversarial phishing websites "from the wild Web" that bypassed a production-grade ML-PWD (reliant on visual similarity) in July 2022. A close inspection shows that these

adversarial pages adopt various evasion strategies such as using blurry logos and adding background patterns (example in Fig. 4.6 in the Appendix). We will use this set (denoted as *APW-Wild*) to examine the user perception on adversarial webpages crafted by real phishers (we note that neither Lee et al. [164] nor Abdelnabi et al. [44] considered real phishing webpages that bypassed a production-grade ML-PWD).

4.3 User Study: Set-up

We carry out two user studies. The first, serving as a baseline, examines how well users can distinguish legitimate webpages from "unperturbed" phishing webpages. The second examines how well users can distinguish "adversarial" phishing webpages (APW) from legitimate ones. Henceforth, we refer to the first user study as *baseline study*, and to the second as *adversarial study*.

4.3.1 Candidate Webpages

Considered Brands. To conduct a meaningful research, we only consider webpages representing a limited set of well-known brands.[2] Hence, we select the 15 well-known brands (typically targeted by phishing attacks [38]) shown in Table 4.1.

Webpage Classes For these selected brands, we construct a user study dataset spanning the following classes of webpages:

- *Legitimate.* For each brand in Table 4.1, we retrieve the (legitimate) webpage corresponding to the brand's homepage.
- *Unperturbed Phishing*. For each brand, we randomly sample two phishing webpages from our chosen dataset (cf. §4.2).
- *APW-Lab*. For each brand and perturbation type, we select one adversarial webpage that bypassed our ML-PWD.
- *APW-Wild*. From the 100 webpages collected by Apruzzese et al. [54], we find 28 of them matching 8 of our target brands (i.e., Apple, AT&T, DHL, Dropbox, Google, Microsoft, Outlook, and Paypal), hence we randomly draw from these 28. We show some examples in Appendix 4.8.

Overall, our user studies entail 15 legitimate, 30 unperturbed phishing webpages, 60 *APW-Lab* webpages, and 28 *APW-Wild* webpages.

4.3.2 Questionnaire Design

Both of our user studies are designed as questionnaires following a similar structure, depicted in Table 4.2. In what follows, we describe this common user study process from a participant's perspective.

General Procedure. At a high-level, the questionnaires consist of three parts. **(1)** A participant starts by reading a consent form stating their rights and the study's objectives. Afterwards, the participant reads a brief introduction about phishing attacks and phishing websites. We explicitly inform the participants that the study is about detecting phishing websites. This is considered a "highly-primed" setting,

[2] Indeed, some users may not be familiar with some less-popular brands, and their responses would have limited value for the purpose of our RQ.

Brands
Adobe, Amazon, Apple, AT&T, Bank of America, DHL, Dropbox, eBay, Facebook, Google, Microsoft, Outlook, Paypal, Wells Fargo, Yahoo

TABLE 4.1: We selected 15 brands, popular in the U.S., for our user study.

Study	Pages Seen by Each Participant	Participants
Baseline	7 Legitimate + 8 Unperturbed Phishing	235
Adversarial	7 Legitimate + 4 *APW-Lab* + 4 *APW-Wild*	235

TABLE 4.2: Summary of our user studies. We report the classes of webpages that *each participant views* and the number of participants.

i.e., participants may be more prepared to detect phishing websites than they would in the real world. We use this setting to estimate the *upper-bound performance* of users. This effect has been shown in previous phishing studies (e.g., [138]) where highly prompted participants have a better phishing detection performance than unprompted participants. **(2)** Then, the participant will view a total of 15 webpages (as screenshots, taken in high resolution and tailored for desktop browsers), covering all 15 brands in Table 4.1. The participant is asked to assess the legitimacy of each shown webpage. For the baseline study, each participant will view 7 legitimate pages and 8 unperturbed phishing pages. For the adversarial study, each participant will view 7 legitimate pages, 4 *APW-Lab* (one for each perturbation type), and 4 *APW-Wild*. The webpages to present to each user are randomly chosen, but we ensure the benign-to-phishing ratio and also that any given user will not see two (or more) screenshots of the same brand—thereby ensuring consistency, since all users will see 15 screenshots of 15 different brands). Furthermore, the order of the pages is randomized for each participant to avoid order bias [114] (this was not done by Lee et al. [164] or Abdelnabi et al. [44]). **(3)** Finally, the participant will answer some exit questions to report demographic information such as age, gender, education, and knowledge of phishing and the considered brands. For attention check, at the end of the main experiment we show a screenshot of a popular social network (Twitter/Instagram) and ask whether it represents a bank website.

Detailed Questions. Under each screenshot, we include two questions: "*How do you rate the legitimacy of this webpage?*" [Q1], and "*What specific components/indicators on the webpage have influenced your choice?*" [Q2]. For Q1, the participant is asked to rate the legitimacy of the web page from 1 to 6: 1 (definitely phishing), 2 (very probably phishing), 3 (probably phishing, but not sure), 4 (probably legitimate, but not sure), 5 (very probably legitimate) and 6 (definitely legitimate). The six-point Likert scale does not include a "neutral" option to encourage participants to draw a conclusion. For Q2, the participant provides open-ended answers via a text box.

For the exit questions, we first inquire the participant's familiarity with the considered brands—"*Do you know these brands/companies/services?*" and "*Please rate how often you visit the websites of these brands*". The participant provides a binary answer for the first question and uses a 4-point Likert scale for the second question. Then, we ask the participant about their gender, age, highest education level, and whether they have a technical background in cybersecurity. More details about these questions are in Appendix 4.9.

4.3.3 Recruitment, Ethics, and Demographics

Our study was reviewed and approved by our IRB; we also follow the Menlo report [75] and do not deploy any phishing webpage on the Web (we only show screenshots). We recruited participants from *Prolific* between July and August of 2023. We choose Prolific over other platforms such as MTurk for the high-quality work from Prolific [204]. Participation in our study is anonymous and voluntary, and participants have unlimited time to read the consent form. Participants can withdraw their consent at any time without any risk. We did not collect any personally identifiable information [155], nor sensitive data [37]. Considering that our target brands are mostly U.S.-based websites, we focus on participants from the U.S. from Prolific. After filtering out low-quality answers (based on attention check), our sample[3] encompasses n=470 participants (235 for each study). The age distribution ranges from 18 to 70+, with 240 males and 220 females (6 non-binary and 4 prefer not to say). Each participant can only join once and receive $2.2 compensation. On average, each participant spent 18.1 minutes on each questionnaire.

4.4 Detection Results (Quantitative)

We first focus on answering RQ1–RQ3. To this purpose, we perform a *quantitative analysis* of the responses we collected for our two user studies. We begin by reporting the results at a high-level (§4.4.1), and then perform formal regression analyses (§4.4.2 and §4.4.3) to assess the statistical significance of our observations.

4.4.1 Overview (how good are our respondents?)

We report the overall performance of both user studies in Fig. 4.3, showing how well our participants correctly recognized each webpage.[4] By comparing the results of the two user studies (useful for RQ1), we observe that our participants exhibit a similar performance in identifying *legitimate* webpages (86% for the baseline study, and 88% for the adversarial study). In contrast, and perhaps worryingly, we found that their ability to recognize *phishing* webpages is much worse; intriguingly, however, it appears that our respondents can more easily discern adversarial phishing webpages (62%) than "unperturbed" ones (51%).

In Fig. 4.4, we focus on the detection rates for *phishing* webpages. Specifically, we break down the results for the *adversarial* phishing webpages (*APW-Lab* and *APW-Wild*) and compare them with the "unperturbed" ones of the baseline study (useful for RQ2). This more detailed comparison reveals that our respondents are not easily tricked adversarial perturbations entailing 'typos' (i.e., the detection rate for *APW-Lab_typo* is 85%). However, they appear to be unable to spot other types of perturbations (i.e., the detection rate for the other three types of *APW-Lab* ranges between [50–56%]). Finally, the detection rate of *APW-Wild* aligns with the general trend (63%), suggesting that adversarial webpages "from the wild Web" are less effective at fooling real users.

[3] Our user studies have a **population that is larger** than most previous user studies on (non-adversarial) phishing webpages [76]. Specifically, most works ([51, 53, 66, 67, 106, 133, 158, 159, 225, 228, 265]) have less than 100 participants, while five ([123, 145, 192, 251, 263]) have [100–400] participants. Only the work by Purkait et al. [216] has more participants (621) than ours, but it was carried out in 2014.

[4] To do this, we take the responses to [Q1] for every screenshot and considering ratings [1–3] as a "legitimate" classification, and ratings [4–6] as a "phishing" one (see §4.3.2).

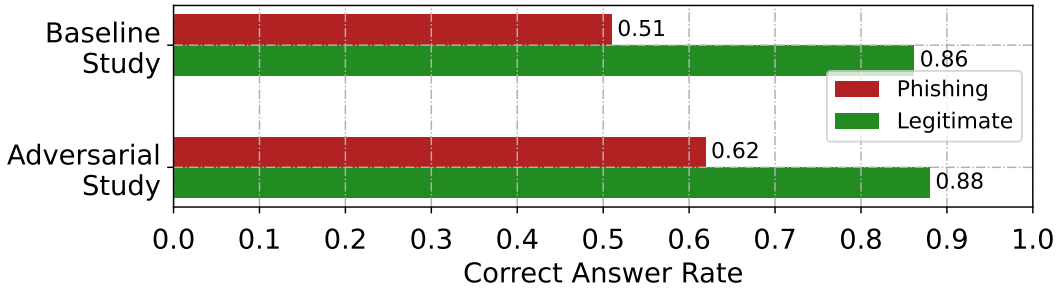


FIGURE 4.3: **Overview of baseline and adversarial study (7,050 responses)**

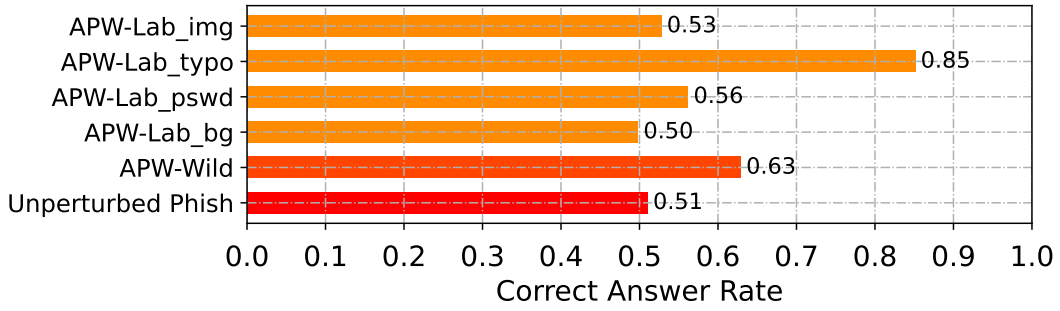


FIGURE 4.4: Detection rate for different types of phishing webpages.

Observations: (1) Our respondents can be deceived by phishing webpages. (2) Some adversarial perturbations are easy to spot by humans. (3) Adversarial webpages from the real world are less effective than "unperturbed" phishing webpages.

4.4.2 Statistical Analysis: Websites (RQ1 and RQ2)

To answer RQ1 and RQ2, we perform a rigorous analysis to ascertain the statistical significance of our previous findings.

Method. We choose a *mixed-effects logistic regression model* (used in many similar studies [76, 265]) to model the process of a user classifying a given webpage. The dependent variable (y) is *the correctness of the user's classification result for this webpage*. The answer is coded as "1" if the classification is correct, and "0" otherwise. We model webpage types and user familiarity with the brand as *fixed effects* (independent variables). We treat each participant as a *random effect* because the same user has viewed 15 webpages (i.e., repeated measures). In this model, we have 3 independent variables (x) related to the webpage: (1) webpage type, (2) the user's prior knowledge of this webpage's brand, and (3) the user's frequency of visiting webpages of this brand. We include (2) and (3) for a simple intuition: if a user is familiar with a brand and visits its webpages regularly, they would be well-acquainted with its typical appearance, and thus are more likely to have a better detection accuracy. For webpage type, we have 7 types, and we treat "unperturbed" phishing webpages as the reference to compare with other 6 types. For knowledge of the website, we code the answer into a binary format and use "No" as the reference. For the website visit frequency, we also code the answer into a binary format and use "Rarely or Never" as the reference.

Variable	Estimate (β)	Std. Err.	p-value
Intercept	0.161	0.146	0.271
Website type: Reference = Unperturbed Phishing			
Legitimate	1.912	0.073	**<0.001*****
APW-Lab_img	0.049	0.144	0.734
APW-Lab_typo	1.723	0.193	**<0.001*****
APW-Lab_pswd	0.185	0.145	0.202
APW-Lab_bg	-0.075	0.144	0.605
APW-Wild	0.484	0.089	**<0.001*****
Knowledge of Website: Reference = NO			
YES	-0.034	0.145	0.812
Frequency of Visiting: Reference = Rarely or Never			
Sometimes or Frequently	-0.169	0.059	**0.004****

TABLE 4.3: **Webpage Classification Analysis** – Logistic mixed-effects regression model: we predict whether a website is classified correctly by a user, based on the type of website, the user's knowledge of this website/brand, and the user's frequency of visiting the website. Statistical significance is denoted by *** ($p < 0.001$), ** ($p < 0.01$), and * ($p < 0.05$) [100].

Results. The model is summarized in Table 4.3. We report standard metrics including *Estimate*, *Standard Error.* and *p-value* for the hypothesis tests. *Estimate* (β) describes the estimated effect of each predictor variable on the dependent variable while holding all other predictor variables constant. The sign of Estimate indicates the direction in which the dependent changes with the independent variables. A positive sign means that as the independent variable increases, the dependent variable also increases; otherwise, the dependent variable decreases. *Std. Err.* represents the average distance that the observed values fall from the regression line. The *p-value* in the regression model describes whether the relationships observed in the samples by chance; usually, the influence was considered statically significant when p<0.05.

Analysis. The results in Table 4.3 confirm our earlier observations from descriptive statistics. First, w.r.t. "unperturbed" phishing webpages, we find that legitimate webpages are statistically significantly easier to detect (β=1.912, p<0.001). Second, among the adversarial webpages, we find two types that are statistically easier to detect by users: *APW-Lab_typo* (β=1.723, p<0.001), indicating that even though the typo is subtle, it has raised suspicion of users; and *APW-Wild* (β=0.484, p<0.001), revealing that while some adversarial webpages from the wild Web can bypass production-grade ML-PWD, they indeed make users more suspicious (w.r.t. "unperturbed" phishing pages). Finally, we did not find statistically significant differences between "unperturbed" phishing webpages and other types of APW. These include adversarial phishing webpages with image footers (*APW-Lab_img*), or visible passwords (*APW-Lab_pswd*), or with changed background images (*APW-Lab_bg*): all these APW can bypass state-of-the-art ML-based detector and yet do not raise more suspicion from users' perspectives.

Table 4.3 also shows an intriguing phenomenon regarding how users' familiarity with the brand correlates with their detection performance. First, we did not find statistically significant evidence that users' prior *knowledge of a brand* influences their detection. However, users' *frequency of visiting the brand's webpages* has a statistically significant *negative* correlation with their detection correctness (β=−0.169, p=0.004).

Variable	Estimate (β)	Std. Err.	p-value
Intercept	0.693	0.018	<0.001***
Gender: Reference = Female Male	-0.001	0.013	0.964
Age: Reference = Younger (<= 39) Older (>39)	-0.004	0.012	0.751
Education: Reference = Lower (< Bachelor) Higher (>= Bachelor)	-0.004	0.013	0.783
Phish knowledge: Reference = NO YES	0.036	0.013	**0.008****
Computer knowledge: Reference = NO YES	0.029	0.019	0.122
Security knowledge: Reference = NO YES	-0.003	0.029	0.931
Time Spent on Survey	-0.001	0.001	0.293

TABLE 4.4: **User Attribute Analysis** – Linear regression model: we predict a user's detection accuracy based on the user's attributes such as demographic factors, technical background, and knowledge of phishing. Statistical significance is denoted by *** ($p < 0.001$), ** ($p < 0.01$), and * ($p < 0.05$) [100].

In other words, users are more likely to make incorrect guesses about webpages of brand that they visit "sometimes or frequently", compared with another that they "rarely or never" visit. This may suggest that familiarity with the brand could lead to overconfidence, i.e., where one's judgmental confidence exceeds one's actual performance in decision-making [210, 256].

TAKEAWAYS (RQ1-2): We make four statistically significant findings. From a user perspective, compared to "unperturbed" phishing webpages: **(1)** adversarial phishing webpages with typo-based perturbations are easier to detect; **(2)** adversarial phishing webpages found in the wild Web are more recognizable; **(3)** adversarial perturbations such as inserting images to the footer, making the password visible, or adding a background image, do not make phishing webpages more suspicious. Finally, **(4)** users are more likely to misdetect webpages that they visit more frequently.

4.4.3 Statistical Analysis: Users Attributes (RQ3)

We now turn our attention to RQ3, and rigorously examine how users' attributes influence their phishing detection performance.

Method. We construct a user model using a *linear regression model* (used in many related studies [76, 216]). The dependent variable is a user's correct answer rate (i.e., accuracy) among the 15 pages they viewed. The independent variables include various user attributes such as demographic factors, technical backgrounds, knowledge of phishing, and time spent on the survey. We code the independent variables in a binary format, except for the time spent on the questionnaire (which is numerical).

Results and Analysis. We display the results in Table 4.4, showing the absence of statistically significant evidence that users' demographic factors affect their phishing

detection performance. Instead, a user's prior knowledge of phishing has a statistically significant influence. More specifically, users with prior knowledge of phishing are more likely to achieve a higher detection accuracy (β=0.036, p=0.008). Even though the estimate β is small, the difference is statistically significant. Our result (in the context of *adversarial* webpages) is slightly different from prior user studies on phishing [117, 138, 158, 216, 240] wherein researchers found that demographic factors such as gender or age have influenced users' detection performance. Finally, the time a user spent on the survey does not seem to have a significant influence on the user's detection accuracy.

TAKEAWAYS (RQ3): We did not find statistically significant evidence that demographic factors affect users' detection accuracy. A user's prior knowledge of phishing is a significant predictor.

4.5 Users' reasoning (Qualitative)

We now address RQ4. Recall (see §4.3.2) that, for every webpage shown in the questionnaire, we also asked (with [Q2]) participants (P) to point out the cues that influenced their rating (of [Q1]). Here, we qualitatively analyze the open-form answers through a *thematic analysis* [245] (which has been used also in [54]).

Codebook. Given that we focus on adversarial phishing webpages, our qualitative coding is based on the data from the adversarial study. In total, we have 3,525 responses from 235 participants from the adversarial study. Two authors (i.e., coders) have worked together to code the answers. A primary coder first codes 27% of the responses, which serves as the foundation for creating a comprehensive codebook. Subsequently, both the primary and secondary coders independently code 10% of the responses that have not yet been coded. We use Cohen's Kappa (κ) statistic to assess the agreement between coders. In cases where κ<0.7, both coders meet up to discuss and resolve discrepancies and refine the codebook, potentially also re-examining and re-coding responses that exhibit inconsistencies. This iterative process continues until a satisfactory agreement is reached, i.e., κ>0.7 [186]. In our finalized codebook, we have κ=0.718, indicating *good inter-coder reliability* [116]. With this codebook (which we release [3]), we thematically coded 1 307 valid responses (37%) that mentioned any webpage elements [54] (e.g., logo, background) or their feeling of the webpage. Specifically, 737 responses are from webpages rated as "phishing" and 541 responses are from webpages rated as "legitimate".

4.5.1 Why is the webpage legitimate/phishing?

We first investigate what led our participants to derive that a given webpage is legitimate or phishing. For the sake of this analysis, we ignore the ground truth of each webpage, since we are interested in the users reasoning of what *they think* is phishing (or not).

"I think this is Phishing because..." Among the 737 responses on webpages rated as phishing, the most prevalent factor is "text content" (282, 38%). Other top-3 factors are "layout" (170, 23%) and "functionality" (168, 23%) of the webpage. Fewer responses (66, 9%) mentioned image content. (We omit factors whose prevalence is below 9%.) We run pairwise Chi-squared tests to compare the number of responses mentioning *text content* (the most prevalent) and those mentioning each of the other

factors. We confirm that the differences are statistically significant (all comparisons have p<0.001).

Among the 282 **text-related** responses, 119 of them (42%) mentioned the presence of typos. For example, P404 stated "*The spelling of the word Outlook is not right*". This is consistent with prior studies [111, 181] reporting that typos hurt the perceived credibility of a webpage. Other text-related responses encompassed factors such as "grammar" (67, 24.5%) and "style" (44, 15.6%). E.g., P1013 mentioned "*The font does not look like the regular Google font that I usually see*".

Regarding other prevalent factors, **layout** (23%) refers to the organization of different components of the webpage, which is a known factor that influences the perceived credibility of websites [78]. E.g., P496 stated "*This does not look like the regular Google login page at all; it looks really off so it seems super sketchy.*" The **functionality** (23%) denotes the specific tasks that the website can help users to accomplish. E.g., P520 mentioned "*This does not appear to be a correct website for DHL since they would not ask you to log in typically to track*". Nonetheless, participants expected that phishing websites would have a way to collect user data. As such, such information-gathering functionality can raise suspicion. E.g., P825, in response to the page shown in Fig. 4.7 (Appendix 4.8), stated "*it asked for the credit card number and therefore looks like it phishing*".

In comparison, fewer responses mentioned **image** content (66, 9%). E.g., P860 mentioned "*The image seems off from what I am usually used to*". Among these, 25 responses mentioned the background, e.g., P1202 stated "*The background isn't moving like on the real site*".

"I think this is Legitimate because..." Among the 541 responses for webpages rated as legitimate, 249 (46%) did not mention any specific factor but describe how the participant "feels" about the webpage. E.g., P154 stated: "*(It) looks like PayPal login page*". Only few responses mentioned specific factors. E.g., 26 (5%) mentioned "*no misspellings or poor grammar*", suggesting that correct writing is regarded as an indicator of legitimacy (albeit this could be influenced by previously viewed webpages having typos). Finally, we report that some users may rely on misinformed strategies. E.g., P54 stated: "*Google is a very reputable and credible search engine*", suggesting that a brand's reputation is an indicator of trustworthiness (which is exactly what phishers use to trick their victims).

TAKEAWAYS (RQ4): After determining the legitimacy of a webpage, users motivate their decision by describing their "feelings" if they believe the webpage to be legitimate. In contrast, when they think the webpage is phishing, they mention more specific indicators—most of which entail textual content errors.

4.5.2 What do users write on adversarial samples?

In an attempt to exhaustively answer RQ4, we further enrich our analysis by performing a break down of the participants' reasoning on the *specific type* of APW (cf. §4.2) included in our adversarial study. For this investigation (and contrarily to what we did in §4.5.1), we must account for the ground truth of each webpage.

APW-Lab. We recall (cf. Fig. 4.4) that our participants performed very well on *APW-Lab_typo*, for which we coded 93 responses. Among these, a large majority (69, 74%) mentioned "typo" (after making a correct detection). Intriguingly, 15% (14) provided reasons that have nothing to do with *APW-Lab_typo* (despite still rating them as phishing). E.g., P668 stated: "*figures do not look normal*". The remaining

11% incorrectly labeled the webpage as legitimate (e.g., "*Everrything looks normal*" [P621]).

Concerning *APW-Lab_img*, we have coded 61 responses. Notably, only 13% (8) pointed out the 'correct' adversarial perturbation (i.e., images on footer). E.g., P544 stated: "*low quality and strange icons at the bottom, which a legit site would not have*". In contrast, 48% (29) mentioned other reasons. E.g., P210 stated: "*Adobe doesn't require logging in to view something in it to my knowledge*". The remaining 39% incorrectly labeled the webpage as legitimate (e.g., "*norton certificate makes me think it's more legit than not.*" [242]).

For *APW-Lab_pswd*, we coded 137 responses. The majority (70, 51%), despite stemming from a correct detection, have nothing to do with our perturbation: only 8% (11) pointed out the visible password as a potential phishing indicator (e.g., "*password field is plain text*" [P1306]; or "*the password is not hidden*" [P937]). The rest 41% incorrectly labeled the webpage as legitimate (e.g., "*As a Wells Fargo customer who was literally just checking their account before starting this study I can assure you this is legitimately legit*" [P86]).

We coded 89 responses for *APW-Lab_bg*. Surprisingly, only 4% (3) of responses mention our inserted perturbation. In contrast, 48% (43) justify their (correct) phishing detection by mentioning unrelated factors. E.g., P971 stated: "*too many big competing brands at the top*". The rest 49% incorrectly labeled the page as legitimate (e.g., P321 stated: "*good grammar, good syntax, appropriate colors, logo*").

For each type of APW above, we again run a Chi-squared test to compare the number of correct phishing detections that mention the inserted perturbation w.r.t. other factors (we do not include misclassifications). The results show that the number of mentions of inserted perturbations is statistically significantly lower than other factors, with $p<0.001$ for all four perturbation types.

> **TAKEAWAY** (RQ4): Even though participants can recognize an APW as "phishing", they rarely pinpoint the perturbation that makes the webpage "adversarial" (as long as it is not text-based).

APW-Wild. We coded 594 for adversarial webpages "from the wild Web". We recall (§4.4) that our participants are better at detecting *APW-Wild* (w.r.t. unperturbed phishing webpages), so we attempt to find an explanation for this. Driven by our previous findings (§4.5.1), we scrutinized whether the reason lies in text-related factors. Among the justifications for correct detections, we found that 22% (131) mention text-related factors (e.g., P1246 wrote "'*Forgotten password' doesn't seem right*"). More specifically, the responses mention typo, grammar, and text-style issues 8%, 6%, and 6%, respectively. Some (18%, 107) mentioned layout (e.g., P362 wrote "*bad css*"), whereas others (16%, 94) mentioned functionality (e.g., P795 wrote: "*(It) should be one form of 2FA*"). Few 9% (56) mention the logo (e.g., P1007 wrote "*The Google logo is wrong.*"); and even less (7%, 40) mentioned other visual elements such as background color (e.g., P108 wrote: "*Google login prompt is not with a gray background*"). Finally, 205 (35%) incorrectly labeled ther webpage as legitimate (e.g., "*Nothing misleading*" [P119]). We run a Chi-squared test, and confirm the number of mentions of text indicators is higher than functionality, logo, and other visual elements, with statistical significance ($p<0.01$ for all pairs). However, the difference between text indicators and layout is not statistically significant ($p=0.082$).[5]

[5]We refrain from making claims pertaining the "correct identification" of the perturbation (as we did for *APW-Lab*): this is because we cannot be sure of which perturbation was applied by the (real) attackers who crafted the webpages in *APW-Wild* [54].

4.6 Discussion

Comparing with Prior Phishing Research. Our work examines how users perceive *adversarial phishing webpages*, which has never been studied in prior works. This provides an interesting data point to contrast with prior studies on generic phishing websites and emails [76]. We discuss four points. **(1)** Prior studies show that men perform better on phishing detection tasks (website [140, 158], email [256, 260]) and a few studies show that women perform better (website and email [202]). Our analysis does not find statistically significant differences among genders (§4.4.3). **(2)** Prior studies show that elders are more susceptible to phishing websites [158, 240]. We again do not find statistically significant differences with respect to age groups (§4.4.3). **(3)** Our study echoes prior results that phishing knowledge correlates positively with users' phishing detection performance [108]. However, surprisingly, we find that the frequency of a user visiting a target brand's website *negatively* correlates with the user's ability to detect phishing webpages targeting this brand (§4.4.3). An explanation is that "familiarity with a brand" leads to overconfidence [210, 256]. This may align with the prior observation that people feel more comfortable with (i.e. trusting) websites that they are familiar with [246]. **(4)** Prior studies have independently shown that typos [117, 181], webpage layout [78], and webpage visual appearance [53] would influence the perceived credibility of websites (and unperturbed phishing webpages). Under the context of *adversarial* phishing, our study shows that participants are significantly more sensitive (§4.5.1) to adversarial perturbations related to typos and text in general (w.r.t. other visual perturbations).

Implications for 'technical' Web Security. For research focused on adversarial phishing attacks (e.g., [60, 98, 164, 171, 229]), we argue that bypassing a given ML-PWD is necessary but not sufficient for a phishing webpage to be successful. The adversarial phishing webpages should be also assessed with users. More importantly, it is important to compare adversarial phishing webpages with unperturbed phishing webpages to ensure the adversarial perturbations do not make the webpages significantly less effective on users (in favor of bypassing ML-PWD). E.g., in our study, we find that certain adversarial perturbations (e.g., typos) are more easily noticed by users despite their high evasion success rate against ML-PWD. This defect would be otherwise unknown without a user study. Another implication is that visual-based adversarial perturbations seem to be effective against both ML-PWD and users, which should be considered in future work when robustifying ML-based phishing detectors. Finally, we stress that some of our visual perturbations were "large" (e.g., *APW-Lab_bg* entailed replacing the entire background—see Fig. 4.2), but they still allowed the webpage to bypass the ML-PWD (both ours and the production-grade one—see Fig. 4.5) *and deceive the users*. This is in stark contrast with most AML research in computer vision, wherein the goal is to apply "imperceptible" perturbations (e.g., [80, 238]). Hence, we endorse future research to explore perturbations having higher magnitude.

Implications to User Education. Researchers have studied ways to improve users' ability to recognize phishing websites through training and education [161, 192, 263]. Our results show that users overlook 'visual' adversarial perturbations (w.r.t. text-based ones). One possible future direction is to increase user awareness of such adversarial phishing webpages. However, we believe there is an inherent risk to train users to search for such visual artifacts. Indeed, adversarial phishing webpages have certain visual artifacts that deviate them from authentic phishing webpages—helping users recognize such artifacts may help users with

phishing detection. However, *the lack of such artifacts* does not mean the website is trustworthy. In our study, we have observed signs of over-trusting known/familiar websites. For example, a user's frequency of visiting a brand's website negatively predicts the user's phishing detection accuracy on this brand.

Limitations. First, our study is limited to participants from the U.S. given we are primarily assessing phishing sites targeting the US-based brands. Future work may consider recruiting participants from different countries and expanding the set of target brands. Second, our evaluation is intentionally set to be highly primed to examine the upper-bound performance of users. This can be different from real-world scenarios wherein users are often "unprepared" when encountering phishing websites. Third, to protect users, we only present phishing screenshots (to prevent users from accidentally clicking on malicious links or leaking their information). However, this also prevents interacting with the website which can be a part of the human's detection process. Furthermore, our screenshots are for desktop browsers, and hence we do not claim that our results generalize to other platforms (e.g., smartphones). Finally, to focus on adversarial phishing webpages, we excluded URLs from our evaluation. Even though prior studies [106, 173, 263] showed that most users cannot effectively utilize URLs as identity indicators of a website, the presence of URLs may help users judge the overall legitimacy of a webpage together with other indicators.

4.7 Conclusion

We present two user studies (n=470) to assess how human users perceive adversarial phishing webpages that bypass ML-based phishing website detectors. We confirm the threat of adversarial phishing webpages to end-users and compare the effectiveness of different types of adversarial perturbations. We argue that *assessing the users' response to adversarial webpages* should be a mandatory step to evaluate evasion attacks in the context of phishing webpage detection. Our work can serve as a benchmark for future research, and we release our questionnaires, codebook, classifiers, and datasets [3].

4.8 SUPPLEMENTARY FIGURES AND TABLES

4.8.1 Number of Experimental Webpages

Our user study involves 15 well-known U.S. website brands. As illustrated in Table 4.5, for each brand, we have 2 high-quality unperturbed phishing pages, 1 legitimate webpage, 4 types of *APW-Lab* pages, and a variable number of *APW-Wild* pages ranging from 0 to 7.

4.8.2 Additional Example Screenshots

Fig. 4.5 presents four adversarial phishing webpages in [54] that evaded production-grade ML-PWD. Fig. 4.6 shows two *APW-Wild* pages used in our study with a weird background pattern and a blurry logo. Fig. 4.7 is an adversarial phishing webpage (*APW-Wild*) that asks for credit card information.

Brand	APW-Lab	APW-Wild	Unperturbed Phish.	Legitimate
Adobe	4	0	2	1
Amazon	4	0	2	1
Apple	4	2	2	1
AT&T	4	7	2	1
Bank of America	4	0	2	1
DHL	4	2	2	1
Dropbox	4	2	2	1
eBay	4	0	2	1
Facebook	4	0	2	1
Google	4	7	2	1
Microsoft	4	4	2	1
Outlook	4	3	2	1
Paypal	4	1	2	1
Wells Fargo	4	0	2	1
Yahoo	4	0	2	1

TABLE 4.5: **Number of Experimental Webpages**

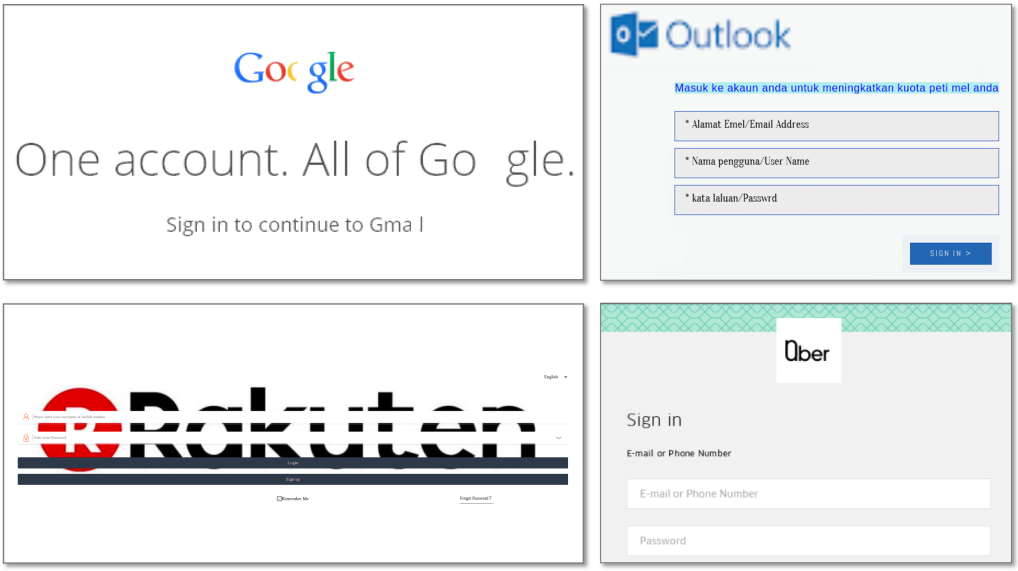
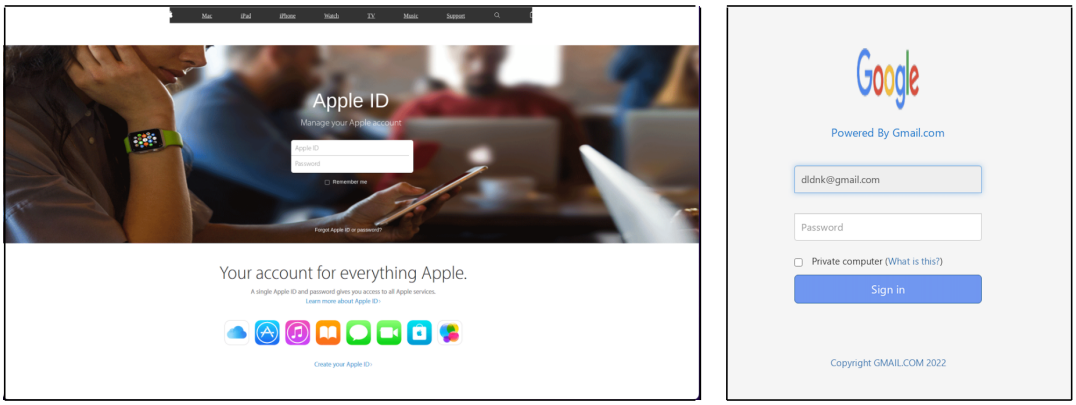


FIGURE 4.5: Four phishing webpages deployed "in the wild" (taken from [54]) which bypassed production-grade ML-PWD.



(a) APW-Wild: weird background pattern (b) APW-Wild: blurry logo

FIGURE 4.6: Additional screenshot of APW-Wild pages used in our user study, to illustrate different adversarial perturbations.

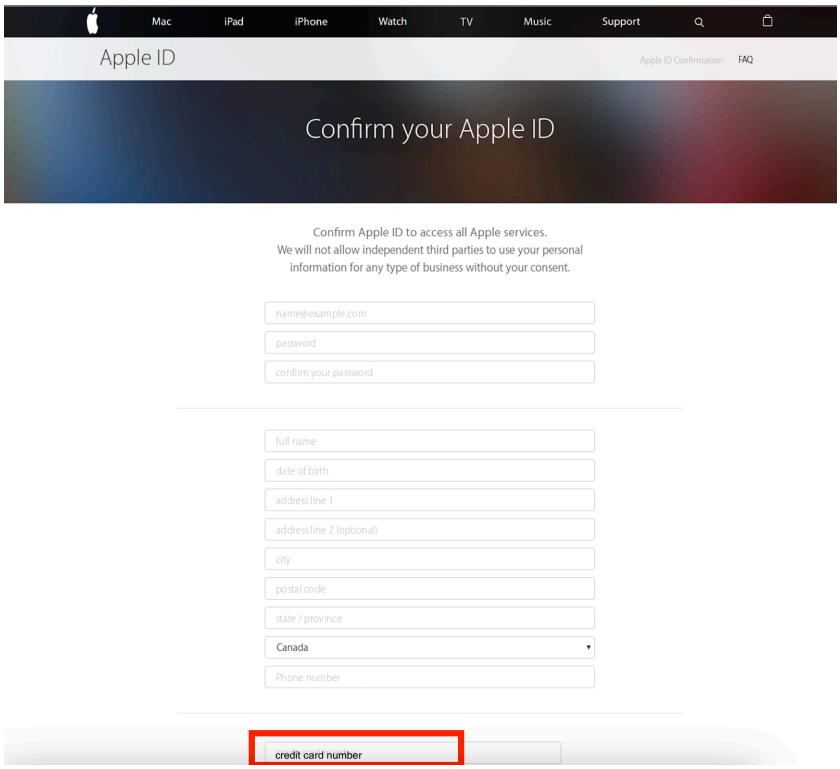


FIGURE 4.7: An adversarial phishing page asking for credit card information.

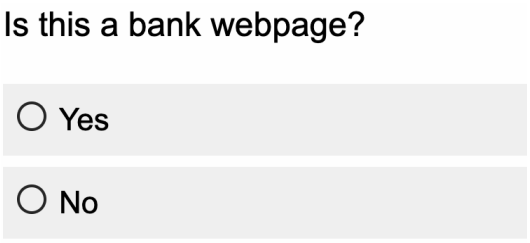


FIGURE 4.8: Attention check question.

4.9 Study Questions

In this section, we show a complete list of our questions, which includes the main task questions and other questions (website knowledge and demographic questions).

Each participant is instructed to review 15 webpage screenshots. Under each webpage, the participant answers 2 questions (15×2=30 questions in total), as shown in Fig. 4.10. Then, we randomly display the screenshot of Instagram or Twitter and show an attention check question (Fig. 4.8). After that, each participant needs to answer 2 questions about website knowledge (familiarity and frequency), as shown in Fig. 4.9 and 6 demographic questions, as shown in Fig. 4.11.

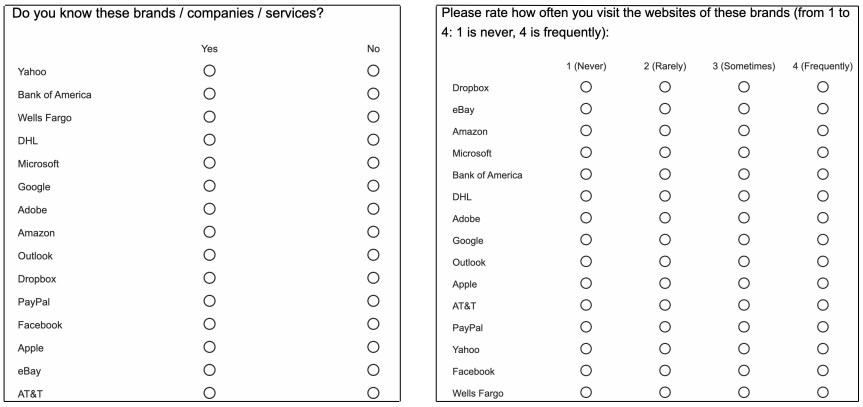


FIGURE 4.9: Other questions: website knowledge.

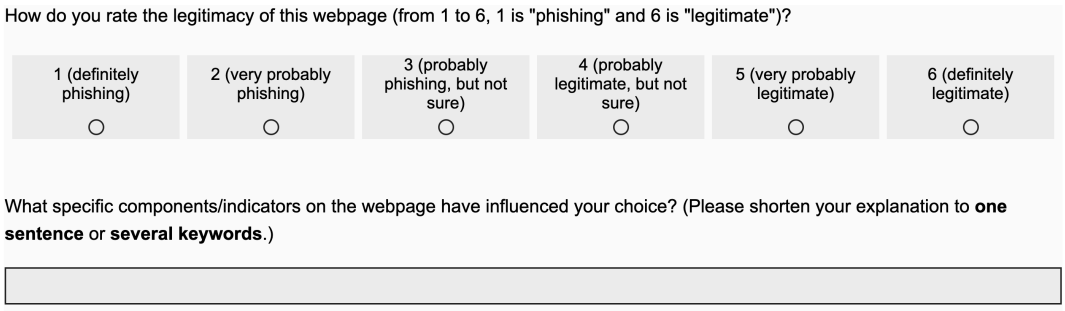


FIGURE 4.10: Main task questions.

Do you have a technical background in cyber security?

- ○ Yes
- ○ No
- ○ Prefer not to say

Do you have knowledge about Phishing websites?

- ○ Yes
- ○ No
- ○ Prefer not to say

Do you have a technical background in computer science or computer engineering?

- ○ Yes
- ○ No
- ○ Prefer not to say

What is the highest level of education you have completed?

- ○ Some high school or less
- ○ High school diploma or GED
- ○ Some college, but no degree
- ○ Associates or technical degree
- ○ Bachelor's degree
- ○ Graduate or professional degree (MA, MS, MBA, PhD, JD, MD, DDS etc.)
- ○ Prefer not to say

How old are you?

- ○ 18-29
- ○ 30-39
- ○ 40-49
- ○ 50-59
- ○ 60-69
- ○ 70 or above
- ○ Prefer not to say

What is your gender?

- ○ Male
- ○ Female
- ○ Non-binary / third gender
- ○ Prefer not to say

FIGURE 4.11: Other questions: demographics.

4.10 Additional Background: Phishing Website Detection and ML security

Phishing websites are a never-ending problem that continue to pollute the Web, and rule-based countermeasures, such as blocklists, cannot cope with such a threat [199]. To provide some form of protection against "novel" phishing websites, modern anti-phishing schemes leverage data-driven techniques [247], such as machine learning (ML). Indeed, thanks to the capability of ML models to "automatically learn from data", it is possible to develop phishing website detectors (PWD) that can identify (and, consequently, block) malicious webpages *before* they are displayed to the end-user—**the actual target of a phishing attack**.

ML-PWD. A large body of scientific literature proposed ML-driven PWD (ML-PWD), which can analyze various data-types to discriminate benign from phishing webpages. For instance, some solutions analyze the underlying HTML of a given webpage [141], or the characters that compose its URL [255], or a combination of the two [60]. Finally, recent approaches rely on deep learning (DL) to compute the visual similarity between two webpages [44], or some of its elements (such as the logo [175]). Due to the promising results of these defenses, *production-grade PWD now integrate some form of ML* to prevent their users from falling victim to a phishing hook [54, 107, 242].

Security of ML. The increasing (and not yet fully understood) successes of ML led to abundant papers to scrutinize its security [80] in adversarial environments. It is now well-known that ML-powered detectors are prone to *evasion attacks*, wherein (tiny) "adversarial perturbations" are added to a given input sample, so as to induce the detector to misclassify it—thereby triggering a **false negative**. Such a vulnerability has been investigated by thousands of research efforts [54], all of which showed that – no matter what – ML models can be easily bypassed (even "adversarially robust" ones [90]). Unfortunately, this problem also affects ML-driven PWD [60, 98, 164, 171]. For instance, some works (e.g. [229]) evidenced that the detection rate of some ML-PWD dropped from 95% to 0 by manipulating just a few features. Moreover, even production-grade ML-PWD exhibit the same weakness: both Google's [171] and BitDefender's [236] anti-phishing schemes have been defeated.

Practitioners viewpoint. Interestingly, however, there is abundant evidence showing that **ML developers do not have the ML-specific weaknesses among their priorities** [54]. Kumar et al. [156] did the first investigation on AML from the perspective of industry practitioners, which indicated only 5 out of 28 organizations had a working knowledge of AML. In the following year, [82] investigated the current state of ML practitioners concerning ML security and privacy, and participants said "I Never Thought About Securing My Machine Learning Models". Even in the latest survey [125], only 28.7% of ML practitioners reported AML knowledge. Simply put, there is a clear gap between AML research and practice, which is not acceptable given the widespread deployment of ML into operational systems. Our paper seeks to rectify this mismatch—which, in the PWD context, presents intriguing properties that are currently overlooked.

Part III

Phishing website detection in multi-language environment

Chapter 5

ChinaPhish: Revealing, Assessing, and Bridging the Gap between Western and Chinese Phishing Website Detection

According to the FBI's 2022 report [20], phishing is the topmost form of cybercrime, whose growth has allegedly increased by over 1000% since 2018. In the second quarter of 2022, the Anti-Phishing Working Group reported over 1M phishing attacks—the worst quarter ever observed [24]. In this context, phishing *websites* are one of the most common vectors employed by attackers, who aim to reach their goals by tricking their victims via apparently legitimate websites [28]. In the first half of 2022, over 200k phishing websites were generated every month [213]—showing that a universal solution to this threat has yet to be found.

The subject of Phishing Website Detection (PWD) is well-studied both in academia and industry. Lots of anti-phishing schemes have been proposed, either "human" centered, such as phishing education (e.g., [148, 157]); or "machine" centered, such as automated detectors (e.g., [134, 222]). This paper focuses on the latter, which do not require any prior knowledge on phishing by potential victims.

Automated PWD can leverage two detection approaches (or a combination thereof), based on either signatures (in the form of "blocklists" [215]), or on data-driven heuristics (e.g., [50, 142, 144]). The former are widely used in browsers; for instance, Google Safe Browsing [32] relies on a constantly updated blocklist which is checked before opening any website, thereby raising an alert if the visited URL is included in such a blocklist. Despite being very precise (i.e., low rates of false positives), blocklist-based PWD cannot detect 'novel' phishing websites [198, 199]. To overcome this limitation, advanced PWD leverage data-driven methods within the domain of Machine Learning (ML): the intuition is to analyze some "features" of a website (extracted from, e.g., its URL or even the underlying HTML [276]) to discriminate benign from malicious webpages. ML-based PWD (ML-PWD) are capable of detecting phishing webpages not included in any blocklist [247], but at the expense of a superior (but still acceptable [55]) rate of false alarms (see §5.1.1 for background).

A large body of literature has shown that – in the right settings – ML-PWD "can work". However, all such papers assumed that the websites (benign or phishing) were in *phonological* languages (e.g., English). Such an assumption, despite being the de-facto standard "in the West", does not allow to determine if (and how much) the proposed solution also works "in the East", i.e., for countries having *hieroglyphic* languages—such as China. In 2016, Li et al. [169] estimated that China suffers 30+B

Yuan in losses every year due to phishing. More recently, the largest Chinese security company [9] reported the yearly trend of phishing attacks *intercepted* in China, which number in the *billions* (shown in Fig. 5.1).

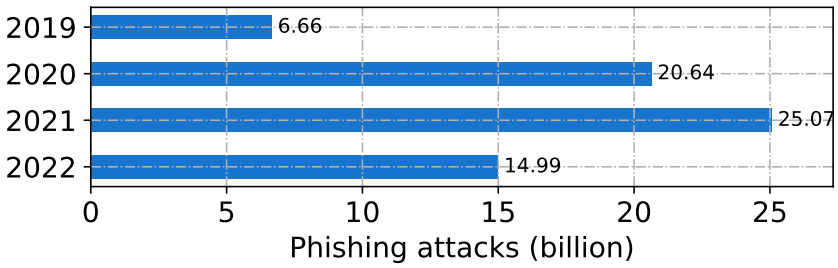


FIGURE 5.1: Phishing attacks intercepted by Qihoo (largest Chinese internet security company) in the first quarters of 2019–2022 [9].

Simply put (as we show in §5.1.2), there is a large side of the World that has been "ignored" by prior research on PWD. Such lack of attention led us to question whether PWD previously shown to be effective for "Western" websites also work for "Eastern" websites—and, specifically, Chinese websites. Indeed, besides obvious differences in languages, Chinese websites present unique characteristics (due to, e.g., some regulations) that set them apart from the rest of the World—and that few papers have considered.

The **motivation** (§5.2.3) of our study has its root in the fact that: *(i)* an increasing number of Western people now reside in China [16]; and that *(ii)* an increasing number of Chinese people migrated to the West [11]. As such, it is important to scrutinize whether PWD can "transfer" between different regions: For instance, an English person can be protected if they live in the UK and only visit English websites—but what if such a person goes to China and visits Western websites? And, vice-versa, Chinese PWD may be effective as long as they are used by Chinese residents—but what if a Chinese person goes abroad?

To answer these questions, we first dissect the anatomy of Chinese webpages, thereby elucidating the major characteristics that differentiate Chinese from Western websites (§5.2). Then, we carry out a large experimental campaign entailing three datasets—each containing thousands of *recent* websites (benign and phishing), taken from *diverse* areas of the World (§5.3). In particular, two are "Western" datasets: one contains *only English websites* [65], while the other contains websites from *various Western languages* (e.g., Italian, French, German, as well as English) [252]; the last dataset contains *only Chinese websites*—which we manually collect and *publicly release*, due to the lack of datasets for Chinese-based PWD available for research. With these datasets, we assess the performance of state-of-the-art PWD (§5.4). We consider *(i)* ML-PWD proposed in research, as well as *(ii)* operational anti-phishing services, and even *(iii)* competition-grade ML-PWD that analyze the HTML of webpages.

Our **results** (§5.5) show that most ML-PWD work well ($\geq$97% F1-score) on websites of the respective "language-group": notably, PWD designed for English websites are effective also for more generic Western websites. However, the performance significantly drops (e.g., $\geq$40% fpr) when a PWD for websites in phonological languages is tested on Chinese websites—and vice-versa. Surprisingly, out of 61 production-grade PWD, *the best one could detect only* 3% *of the Chinese phishing webpages in our dataset*. These results highlight that existing PWD (especially commercial ones!) respond poorly when processing websites of different languages. Based on

these findings, we advocate the need for novel PWD that work under the assumption that the user can visit either phonologic- or hieroglyphic-based websites. Finally, as a constructive step forward, we propose (and empirically evaluate) some ways to address the problem we brought to light (§5.6).
CONTRIBUTION. Our efforts *reveal the gap* between Chinese and Western PWD. We are the first to (ttbook):

- elucidate the differences between Chinese and Western websites (from a PWD perspective);
- assess the "cross-language" effectiveness of state-of-the-art PWD (real systems and research proposals);
- collect and publicly release a dataset for Chinese-based PWD, containing 1 620 websites (from 2022) and provided with their URL, HTML and screenshot.

As a *technical* contribution to help building a bridge to close this gap, we publicly release of entire codebase—including those for our proposed countermeasures [2]. We also provide **additional technical details, original experiments, and thorough analyses** in our Appendix.

5.1 Background and Related Work

We summarize the problem of PWD (§5.1.1). Then, we survey the phishing landscape in China (§5.1.2), highlighting the limitations of related research.

5.1.1 Phishing Website Detection

Phishing is a historical security problem, which has been tackled by abundant research. Reliant on social engineering [84], used to "lure" a victim onto a malicious webpage, phishing website attacks require the victim to *(i)* be shown the webpage; and *(ii)* be "caught" by providing some sensitive data, or clicking on a harmful link [230]. Clearly, the attack fails if the potential victim recognizes the webpage as malicious. Therefore, anti-phishing training programs can reduce the risk of phishing attacks [146], and some research has been carried out (e.g., [160, 224]). However, according to ProofPoint's 2022 report [28], more than 40% of organizations do not deliver adequate training, and nearly 15% almost do not educate workers at all. Hence, there is still a need for "machine-based" mechanisms that provide a first line of defense against phishing for uneducated (or distracted) users. These automated detection schemes are based on the combination of signatures (i.e., *blocklists*) or data-driven heuristics—among which, many rely on *machine learning* methods.

Signature-based PWD. Signature-based PWD still represents the preferred countermeasure against phishing, and leverage blocklists of "suspicious" websites (taken from, e.g., PhishTank [26], or Google Safe Browsing [211]). Before rendering any given website, the browser (or an organization-wide detector) checks if the URL (or a subdomain) is included in the blocklist, thereby alerting the user upon a correct match [276]. To avoid triggering annoying false alarms, the websites in these lists must be verified: as a result, signature-based PWD have high precision—which is appreciated in the context of PWD, given that web-users visit hundreds of pages every day [46]. Unfortunately, signature-based detectors are useless [61] against "novel" attacks. Despite huge efforts put in by the maintainers of blocklists

to keep them as up-to-date as possible, some websites are bound to evade blocklist-based PWD [247]. Such a shortcoming led to the proliferation of complementary PWD that can cope with the ever-changing landscape of phishing websites—which can be accomplished via machine learning (ML).

ML-based PWD. The underlying principle of machine learning is to have "machines that autonomously learn from data." This process is done by *training* an ML *model* over some *training data* by means of a given learning *algorithm*. The successes of ML in various fields (most notably computer vision and natural language processing [150, 163]) showed the remarkable performance of ML for classification tasks, inspiring researchers to investigate their effectiveness also for cyberthreat detection [61]—which also encompasses PWD [71, 95, 119, 141, 183, 233]. Existing ML-PWD can fall into three categories, depending on the information used as basis to perform the (binary) classification of a given website [59]. Specifically, an ML-PWD can use either the URL of a website, its representation (e.g., the image or the HTML), or their combination. Each of these can be elaborated in diverse ways: for instance, some ML-PWD necessitate some preprocessing aimed at extracting some features from a given piece of data (e.g., computing the length of the URL [188]); others (typically those relying on deep learning [44]) may analyze a given input in its raw form. We stress that – despite the appreciable results shown in research – recent findings showed that commercial PWD using deep learning can be easily evaded (by real attackers!) via decade-old tricks [55]. (Additional details are in Appendix 5.12).

Narrow Scope. Despite dozens of papers proposing ML-based countermeasures to phishing, prior efforts (e.g., [50, 65, 68, 142, 162, 189, 203, 247]) only focused[1] on Western websites—overlooking that phishing is a long-standing problem also in other areas of the world, such as China.

5.1.2 The Chinese phishing landscape (in research)

According to the largest Internet security provider in China, Qihoo 360 [9], over 25 *billion* phishing websites were intercepted in the first quarter of 2021—a rate of 280 million per day (cf. Fig. 5.1). Reports estimated over one *billion* Chinese netizers as of June 2022 [14]: accordingly, 24% had been scammed by phishing websites in the previous 6 months. The yearly cumulative losses of Chinese users due to phishing exceed 20B Yuan (~3B USD) [167].

Shortcomings of prior research

Phishing websites are clearly rampant also in China [177]. Yet, this threat is vastly understudied from a research perspective. Almost 10 years ago, in 2014, Zhang et al. [270] proposed 5 domain-specific features to detect phishing websites targeting Chinese eCommerce: despite achieving 96% accuracy, [270] only focus on eCommerce websites, neglecting the plethora of other websites (e.g., forums, hospitals and government) which can very well be targeted by phishers. Indeed, Chinese websites tend to have different characteristics, as shown in Fig. 5.2: eCommerce websites must report their business licence (red box in Fig. 5.2a), which is not necessary for Chinese government websites—which, in turn, have a government identification code (blue box in Fig. 5.2b). More recently, in 2020, Li et al. [167] proposed five space transformations to generate new features that disentangle the linear and non-linear

[1] Some works (e.g., [121]) claim to propose "language independent" PWD. We argue that such a term is misleading: such PWD are affected by the shortcomings of "target dependent" PWD (see Appendix 5.12).

interactions between features in malicious URL data. The dataset used in the experiments of [167] includes URL from generic Chinese websites, thereby allowing one to assess the effectiveness of ML-PWD analyzing the URL of a website. Even though such an approach may work when analysing Chinese websites, its effectiveness is questionable when Western websites are taken into account. This is because the *URL is a combination of alphanumeric characters*: hence, the URL for Western and Chinese websites may appear similar despite their HTML being significantly different; plus, PWD analysing the URL can be easily bypassed (e.g., [65]). We argue that HTML may be a more valuable source of information for PWD; however, as we will show (in §5.2.1), proper usage of the HTML for Chinese websites may require some tweaks that are neglected by ML-PWD previously proposed for Western websites.

(A) Footer of a Chinese eCommerce website (`https://global.jd.com/`).

(B) Footer of a Chinese govt. website (`https://banshi.beijing.gov.cn/`).

FIGURE 5.2: Chinese eCommerce and government websites have different identifiers. Red boxes denote the "business license", whereas the blue box the "government identification code".

Chinese PWDs (Related work)

We discuss the research endeavours on Chinese ML-PWD of the last decade, for which we provide an overview in Table 5.1. For each paper,[2] we report: whether the experimental dataset is publicly *available* (✗ or ✓) and whether it also included websites from different *regions* than China (✗ or ✓); the *focus* of the ML-PWD (either 'generic', or for specific types of websites); the *date* (i.e., year) on which the data was collected; the types of *features* used for the analysis (F_u=URL only, F_h=HTML only, F_c=URL+HTML) and whether these features entail *Chinese-specific* characteristics; and if the conclusions are drawn after making *statistically significant* comparisons (✗ or ✓).

As we can see from Table 5.1, some work [96, 167, 271] only consider ML-PWD analyzing the URL, thereby failing to capture the additional information provided by the HTML (which, as we show in our paper, plays a crucial role). The authors of [274] devise a ML-PWD analyzing HTML features that consider Chinese-specific word embeddings—which are clearly language dependent and (as we show in the next section) are inappropriate for analyzing Western websites. The ML-PWD

[2]We performed an extensive literature search through various search engines, and also looked at cross-references and citations. To the best of our knowledge, Table 5.1 represents the state-of-the-art.

TABLE 5.1: Papers on Chinese PWD. None of these release the source-code (the online tool in [167] is not functional anymore).

Paper (1st Author)	Year	Dataset Available	Other Regions	Website Focus	Collection Date	Analyzed Features (F)	Chinese Specific	Stat. Sign.
Chu [96]	2013	✗	✗	eCommerce	2012	F_u	✗	✗
Zhang [270]	2014	✗	✗	eCommerce	2014	F_c	✓	✓
Zhang [271]	2016	✗	✓	generic	2011	F_u	✗	✗
Li [169]	2016	✗	✗	generic	2015	F_u,F_h	✗	✗
Zhang [273]	2017	✗	✓	generic	2017	F_c	✗	✗
Zhang [274]	2017	✗	✗	generic	2017	F_c	✓	✗
Hu [262]	2017	✗	✗	finance	2017	F_u,F_h	✗	✗
Zhang [113]	2017	✗	✗	finance	2016	F_c	✓	✓
Li [167]	2020	✗	✓	generic	2020	F_u	✗	✗
Liu [176]	2021	✓	✓	generic	2018	F_c	✗	✗

proposed by [273] are assessed on both Chinese and English websites, achieving over 95% accuracy, but the corresponding evaluation is *(i)* not reproducible, and *(ii)* lacks statistical validation—both of which are shortcomings affecting most papers (i.e., [96, 167, 169, 262, 271, 274]) in Table 5.1. As a matter of fact, our experiments will reveal substantially different results than those reported by [273]. Notably, [270] proposed five Chinese-specific features and construct ML-PWD analyzing both the URL and HTML of a webpage, but they only focused on Chinese eCommerce websites (similarly to [96]). Such a narrow focus also affects the research in [113] and in [262], whose proposed ML-PWD are assessed only on financial websites. The recent work by Li et al. [176] proposes a complex PWD, but their dataset includes *only 51 Chinese webpages*, which cannot represent the landscape of phishing in China (and, unfortunately, the source-code of [176] is not provided).

OUR GOAL. Prior research on Chinese PWD is scarce and presents limitations (e.g., lack of statistical validation, reproducibility, or generality). We aim to overcome all such shortcomings and provide reliable results to assess the state of Chinese w.r.t. Western PWD.

We focus on phishing *websites*: papers on other forms of phishing (e.g., email [120, 132, 220]) are orthogonal to ours.

5.2 Western vs Chinese websites

As our first contribution, we elucidate the differences that set Chinese websites apart from Western ones. These differences lie in: the *language* itself; and the *structure* of the website. Both of these influence the representation of the website (and, in particular, its HTML), thereby suggesting that ML-PWD analyzing such information[3] are likely to respond differently on websites of different regions.

5.2.1 Chinese & Western texts

Context. English or Western languages (e.g., German) are phonetic languages. Their smallest sememe words [197] are a combination of 26 alphabet letters. For instance, a generic English word (e.g., "hello") can be easily pronounced with the help of its glyphs. However, Chinese texts (and other Eastern texts, e.g., Japanese)

[3]This section focuses on ML-based PWD because the effectiveness of blocklist-based PWD entirely depends on the "entries" contained in the blocklist, and is hence agnostic to the languages of such entries.

are more complex: there can be little or no correlation between the pronunciation and the glyph of a given word.

Example: The Chinese word '参' has multiple pronunciations: 'cān', 'cēn', and 'shēn'. However, even native speakers cannot determine how to pronounce '参' just by observing its glyph.

Chinese is both a kind of hieroglyphics and phonetic language, which has three unique linguistic characteristics: *pinyin*, *glyph* and *tone* [178]. Only the simultaneous knowledge of these three can uniquely determine a Chinese character, all of which are indispensable. As shown in Fig. 5.3, words in group (a) have the same *pinyin* and *tone*, but their *glyph* and semantics are different, which means the Chinese texts cannot be confirmed only by the pronunciation. The words in group (b), have the same glyph, '长', but they differ in the *pinyin*, *tone* and semantics. Finally, words in group (c), '离' and '里' have the same *pinyin*, 'li', but different *tones* ' ˊ ', ' ˇ ' and different *glyphs*; furthermore, their semantics are different.

FIGURE 5.3: The combination of Chinese texts: *pinyin*, *glyph* and *tone*.

In practice. The difference between Chinese and Western languages is likely to affect the effectiveness of PWD "trained" on either of these languages. For example, many ML-PWD (e.g., [130, 170]) analyze a feature that denotes whether the website's title includes the domain of its URL. Let 'H_titBr' denote such a feature: we represent this extraction procedure for Western and Chinese websites in Fig. 5.4. For a Western website, we (step 1) get the domain from the URL and (2) get the title from the HTML, then (3) check if the title includes the domain. However. for Chinese websites, the title is in Chinese hieroglyphs: hence, we (4) need to 'convert' the title to its corresponding pronunciation, e.g., *pinyin*,[4] a combination of letters; and then (5) compare it with the URL's domain[5].

This difference also exists between Chinese and Western versions of the *same* website. As an example, consider Amazon, for which we provide an illustration in Figs. 5.5. The URL of the Western variant contains the string "amazon", which also appears in the title (as HTML) of the webpage (Fig. 5.5a). Therefore, the extraction of the 'H_titBr' feature (i.e., steps 1, 2, 3 in Fig. 5.4) is straightforward—and this is done also by open-source ML-PWD (e.g., [59]). However, this extraction procedure does not work on the Chinese version of Amazon. As shown in Fig. 5.5b, the HTML's title tag is "亚马逊-网上购物商城：要网购，就来z.cn!", which clearly does not include the string "amazon" (which is present in the URL). Therefore, to *correctly* extract this feature, it is necessary to convert the title to its pronunciation. Not doing so (and applying the 'straightforward' Western procedure) leads to a mismatch that induces an ML-PWD to believe that the Chinese version of Amazon to be a suspicious website.

[4] Among the top30 popular Chinese websites, 15 use pinyin (2023).

[5] A potential workaround would be to apply a different feature extraction process depending on whether the website is in Chinese or not, but we are not aware of phishing detectors that do this. Plus,

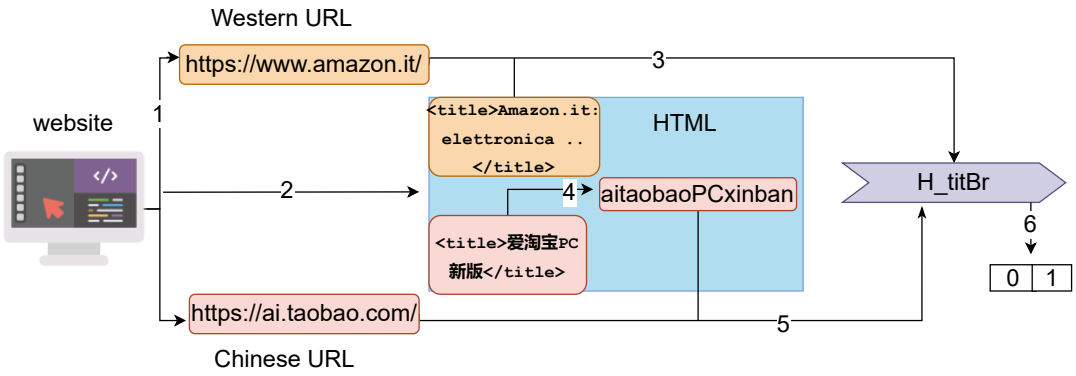


FIGURE 5.4: Extracting 'H_titBr' from Chinese and Western websites.

```
<title>Amazon.com. Spend less. Smile more.</title>
```

(A) Title tag (HTML) of Western Amazon. (https://www.amazon.com/).

```
<title>亚马逊-网上购物商城：要网购，就来z.cn!</title>
```

(B) Title tag (HTML) of Chinese Amazon. (https://www.amazon.cn/).

FIGURE 5.5: Comparison between the URL and HTML title tags.

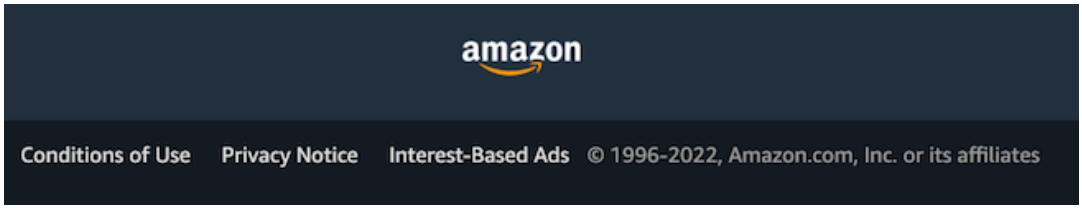
5.2.2 Chinese & Western websites structure

Context. According to China's network security law and the Administration for Industry and Commerce regulations [4], all Chinese websites need to be registered with the Ministry of Industry and Information Technology of the Chinese government (i.e., ICP records). Chinese websites engaged in different activities must apply for qualification certificates from the corresponding government departments. E.g., "JD.com" is an eCommerce that mainly sells electronic merchandise and medicines, thus it received a telecommunication business license from the Chinese Ministry of Industry and Information Technology, and a qualification certificate for pharmaceutical services approved by Beijing Municipal Medical Products Administration (an example is shown in the red box in Fig. 5.2a). In addition, it is common for Chinese websites to display trusted website certifications issued by third-party organizations (or the police) to increase their credibility (see the yellow box in Fig. 5.2). However, Western websites do not have (nor require) these certificates. Even world-renown websites lack them (see Fig. 5.6).

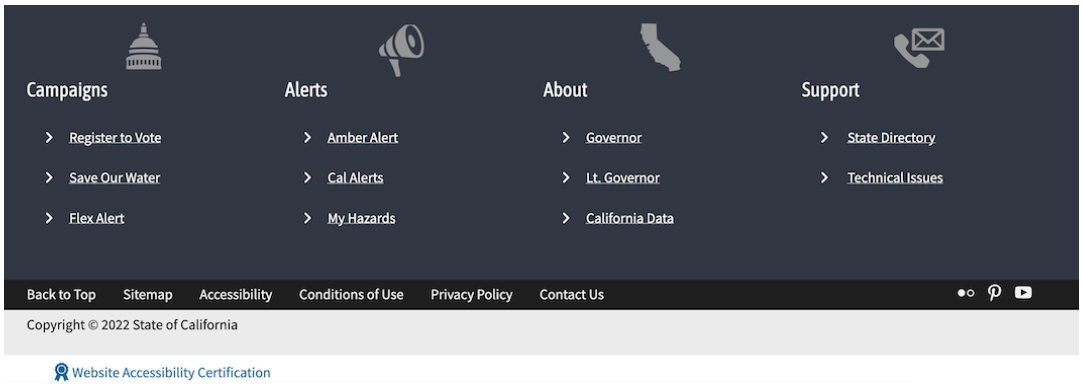
In practice. Analyzing the ICP record is a well-known method to identify malicious websites in China [270]. However, the absence of this form of "certification" for Western websites creates an *intrinsic incompatibility* between Chinese and Western ML-based PWD that analyze the HTML of a webpage. For instance, ML-PWD for Chinese websites will search for the ICP record on Western websites, but will never be able to find it—thereby inducing the ML-PWD to believe that any Western webpage is "suspicious". In contrast, ML-PWD for Western websites will also be adversely impacted by the presence of the ICP record: it is well-known [142, 187, 266] that phishing webpages have many objects that point to "external" items—and, of course, the ICP embedded in a Chinese website points to an external resource. As a

such a verification step would increase the overhead to analyze the page: to prevent trivial spoofing, its authenticity should rely on third-parties.

result, ML-PWD for Western websites will also be more likely to be "suspicious" of any Chinese website.



(A) Footer of a Western eCommerce website (https://www.amazon.com/).



(B) Footer of the Califonia government website. (https://www.ca.gov/).

FIGURE 5.6: Exemplary eCommerce and Govt. 'Western' websites.

5.2.3 Motivation and Research Questions

Problem Statement. Prior research has shown that existing ML-PWD work well on Western websites; and some papers also showed that ML-PWD can be tailored for Chinese websites with some success (§5.1.1). However, as we discussed, Chinese websites are different from Western websites. Such a difference led us to ask ourselves: *How do PWD that are effective on Western websites perform on Chinese websites (and vice versa)?* To the best of our knowledge, previous research (as we showed in §5.1.2) cannot provide an answer to our question, since *(i)* Chinese and Western websites have been mostly treated independently; and/or *(ii)* the few works that consider both "regions" simultaneously have limitations.[6]

A real-world problem. Plenty of Chinese people live in the West, and many Westerners live in China [11, 16]. People living abroad need to browse their home country's website. If Chinese PWD work poorly on western websites, then Westerners who live in China will be more likely to fall victim to Western phishing websites, and vice-versa. Even though it is well known that the Great Firewall prevents [43, 135] Chinese residents from accessing popular Western websites (e.g., Facebook), hence implicitly providing some form of protection to Chinese users against Western phishing websites, some can still be reached (e.g., GitHub). Moreover, usage of VPN services can bypass the Great Firewall, thereby allowing Chinese residents to access any[7] website—but this will expose them to Chinese phishing websites. Indeed, as of Feb. 2023 [27], Google Chrome is the most popular browser even in China (with a share of 56%); however, the anti-phishing filters of Chrome are not tailored for Chinese phishing websites (as we show in §5.4.2). In turn, also users

[6] Actually, the results of [176, 273] may suggest that these two regions are "compatible" from a phishing detection perspective!

[7] These users can even fall victim to phishing in the dark web [268].

who live outside China (either Westerners or Chinese expats) and use Chrome (or similar browsers) can fall victim to phishing Chinese websites. Simply put, if PWD exhibit a poor compatibility between Chinese and Western websites (as our analysis in §5.2 suggests), then many people can fall victim to phishing attacks that (perhaps inadvertently) exploit such a vulnerability.

Research Methodology. Inspired by this dilemma, we want to verify if the gap between Chinese and Western phishing website detection truly exists; and, if so, potentially assess its *impact on real systems*. We tackle this problem by answering three research questions (RQ):

RQ1: Do ML-PWD proposed in research for Western websites work equally well on Chinese websites?

RQ2: Do ML-PWD tailored for Chinese-websites work well when analyzing Western websites?

RQ3: Do closed-source and commercial PWD (either blocklist- or ML-based) typically used "in the West" work well when analysing Chinese websites?

Since Western websites is a broad term, we enrich our RQs by differentiating (a) 'English-only' websites from (b) 'generic' Western websites (e.g., Italian, German, *and* English). Before we do any of the above, however, we have to deal with a crucial problem: the lack of publicly available data for Chinese-focused PWD.

5.3 Data Collection

Answering any of our RQ requires experiments which entail *(i)* assessing the effectiveness of PWD on *(ii)* Chinese and Western websites. Unfortunately, we were not able to find *any* existing resource that provided a representative dataset of the Chinese phishing website landscape. The datasets used by *prior research are not publicly available*,[8] and the (few) *existing repositories only contain lists of URL*,[9] which are outdated or no-longer active and hence do not allow to assess the effectiveness of PWD that analyze information not derived from the URL. In contrast, publicly available datasets having websites appropriate for Western PWD are more complete.

Hence, as a first step towards answering our RQ (and as an additional contribution), we manually collect a dataset that enables the assessment of PWD on Chinese websites (§5.3.2). Then, we explain the procedures and tools adopted to collect the non-Chinese datasets used in our evaluation (§5.3.3). We provide a summary below (§5.3.1).

5.3.1 Overview and Design Choices

Our experiments focus on empirically studying the gap between Western and Chinese PWD. However, recent surveys reveal that English is the global and most spoken language [1], as well as being also the most commonly used website content language (with a share of over 57% [40]). To account for the predominant usage of English on the Web, we must consider it as a stand-alone 'population' w.r.t. other Western languages. Besides allowing for a bias-free evaluation, such a design also

[8] Though Liu et al. [176] released their dataset, it only has 51 Chinese phishing samples which are not enough for a comprehensive analysis.

[9] E.g., there are only 221 phishing websites and no screenshot in [17]; whereas the majority of the entries in [6] are outdated.

allows us to ascertain whether PWD perform similarly across websites using *different phonetic western languages*.

Therefore, for our experiments, we collect (and publicly release) three datasets for ML-PWD: a Chinese website dataset (`ChiPhish`), an English website dataset (`EngPhish`), and a Western website dataset (`WstPhish`). We report an overview of our datasets in Table 5.2, showing their distribution (in terms of benign and phishing samples) and usage in prior research.

TABLE 5.2: Summary of datasets used in our evaluation.

Dataset	#Benign	#Phish	Used in
`WstPhish`	4 269	6 935	[59, 252]
`EngPhish`	11 019	4 092	[65]
`ChiPhish`	1 055	565	(new)

5.3.2 Chinese phishing website dataset (NEW)

Among the contributions of this paper is the first dataset for Chinese-focused PWD, `ChiPhish`. To understand *why* such a contribution is significant, let us describe the difficulties we encountered during the creation of `ChiPhish`.

Challenges

Although finding legitimate Chinese websites is trivial, finding *active Chinese phishing websites* is difficult. For instance, even popular phishing tracking services (PhishTank and OpenPhish) hardly report websites from China—likely because their userbase does not visit Chinese websites. Furthermore, [12] also mentioned that their data under-represent the amount of phishing occurring in China, and they did not collect any attacks against the four largest Chinese banks and major Chinese eCommerce companies. Finally, [5] indicates that more than half of malicious gTLD registrations worldwide were being made by Chinese phishers, and that *six of the top ten* registrars of malicious phishing domains were located in China and had primarily Chinese customers; this data was contributed by APAC which works with phishing targets in China. However, because of the Chinese cybersecurity law [4, 18], APAC or other public cybersecurity platforms in China that were used in previous research no longer broadcast Chinese phishing data anymore. These difficulties may partly explain why the landscape of Chinese PWD has been mostly unexplored in research.

Goals and Creation

We created `ChiPhish` with a twofold goal: (i) enable a meaningful analysis *for this paper*; and (ii) provide a solid foundation *for future work*. Hence, `ChiPhish` is meant to fulfill three requirements:

- *Generality*. It must include websites (benign and phishing) of various type (e.g., not only eCommerce).

- *Representativity.* It must have a sufficient (≥1000) amount of *recent* (≥ 2020) samples[10] (i.e., websites).
- *Completeness*. Samples must be provided with three formats of raw data: URL, HTML, and screenshot.[11]

Our `ChiPhish` dataset has 1 055 benign and 565 phishing samples. For the *benign* data, we relied on Chinaz [31], a popular [168] trusted source which provides a ranking of popular Chinese websites (similarly to Amazon's Alexa rankings). Specifically, our benign samples are taken by using the top-60 websites reported by Chinaz (at the end of 2022) and scraping the links contained in these websites (which we manually verified pointed to trusted websites). As for the *phishing* data, we had to draw from various sources. We searched across the Threat Intelligence Centers of Chinese IT companies (e.g., VenusEye [41], QiHoo 360 [9]), competition platforms and repos [17] and security forums (e.g., kafan [33]) to retrieve hundreds of Chinese phishing websites—which we manually checked to ensure that they were still online. All phishing samples have been verified by the publishers of the respective source (and by ourselves). To the best of our knowledge, `ChiPhish` is the only publicly available dataset for Chinese PWD with these characteristics. A summary of `ChiPhish` is in Table 5.9 (in Appendix 5.9).

5.3.3 Datasets for phonological languages

The research community can benefit from many datasets for "western" PWD, i.e., having phonological languages. However, *most of such datasets do not enable* an evaluation that can provide a satisfying answer to our three RQ. For example, the dataset used by [193] only includes the URL of its websites, thereby preventing retrieval of any data on the corresponding HTML (phishing webpages are taken down quickly) at a later time. The same problem affects the dataset proposed by [44], which reports the screenshot but neither the URL nor the HTML of its samples—and, as discussed in §5.2, the HTML is of crucial importance for our study.[12] Finally, the well-known datasets proposed by [130] and [189] are only provided as pre-computed features, thereby preventing to retrieve the original information on the corresponding website.

After surveying the few existing datasets that provide complete information on each sample contained therein, we observed that all such datasets contained websites in multiple languages. Hence, we had to devise a way to identify the main language of a given webpage. For this purpose, we develop an original Language Selector Tool, LaSeTo (§5.3.3). Moreover, to remove bias, we create our two other datasets by using LaSeTo to extract subsets of two *different* (and recent) datasets which have been *validated by the research community*: the one in [252] for `WstPhish` and the one in [65] for `EngPhish`.

LaSeTo description

Our Language Selector Tool fosters two elements of the HTML alongside Google's Compact Language Detector v3 (CLD3), i.e., an open-source system that

[10] Of course, we do not claim that our dataset will always be representative of the entire Chinese phishing landscape!

[11] Screenshots facilitate research on visual PWD (e.g. [44, 239]).

[12] The PWD in [44] is based on image similarity, which despite showing good results *in research*, are trivially evaded *in practice* [55].

leverages state-of-the-art ML techniques for language identification, supporting over 100 languages [122]. Specifically, LaSeTo considers: *(i)* the 'lang' HTML attribute which is used to declare the language of a webpage; and *(ii)* the language used in the HTML 'title' tag—which can also suggest the primary language of the userbase of a given website. Practically, LaSeTo receives the raw HTML of a webpage as input: if it can detect the 'lang' attribute, it will output the corresponding language; otherwise, it will query CLD3 with the 'title' tag, and provide the corresponding language as output. We release the source code of LaSeTo [2].

The `EngPhish` dataset

Since we want an English-only data corpus, we chose the *latest* suitable dataset as a starting point. Specifically, we consider the dataset provided in the 2022 paper from Apruzzese and Subrahmanian [65], which contains nearly 24k samples (16k benign and 8k phishing). However, not all of these are English websites: we hence submit all 24k samples of [65] to LaSeTo, finding that 15 111 are in English (specifically, 4 092 phishing and 11 019 benign). Overall, these samples represent `EngPhish`. Furthermore, we use `EngPhish` to *verify the accuracy of* LaSeTo. We randomly sampled 100 webpages from `EngPhish` and manually checked their language: only 8 of them were not in English—which we consider as an acceptable margin of error.

The `WstPhish` dataset

Our last dataset should include webpages representing a broad coverage of "western" languages. Hence, we use LaSeTo to extract a subset from the websites provided in the 2021 paper by Van Dooremal et al. [252]. This data corpus entails almost 4M websites, of which 100k are phishing (taken from various repositories); some samples are 'blank' webpages, which we ignore. To create `WstPhish`, we begin by considering 17 phonologic languages from the list of most common [34] European languages (besides English), i.e.: German, Italian, French, Swedish, Polish, Spanish, Norwegian, Hungarian, Czech, Danish, Dutch, Greek, Turkish, Slovenian, Croatian, Romanian and Luxembourgish. Then, we run LaSeTo on the websites in [252], saving all webpages that match any of these 17 languages. We thus obtain 4 269 benign and 6 935 phishing webpages, which represent our `WstPhish` dataset. Finally, recall that our LaSeTo has some margin of error: we can expect that some websites in `WstPhish` may be in other languages—and, likely, in English. Hence, we randomly sample 100 webpages from `WstPhish` and manually check their language: we find that 2 are in English. We conjecture that this small fraction of English samples will improve the generality of the findings derived by analysing `WstPhish`.

5.4 Experimental Testbed

We present the experimental setup to answer our three RQs. We first explain the state-of-the-art ML-PWD that we develop by following prior work (§5.4.1), then we describe the *closed-source* PWD we considered (§5.4.2), and conclude by summarizing the evaluation workflow (§5.4.3).

5.4.1 State-of-the-art ML systems for PWD

We recall that RQ1 and RQ2 entail assessing whether state-of-the-art ML-PWD exhibit similar performance across both Chinese and Western websites. Hence, we

replicate existing ML-PWD which analyze the feature representation of a given website [188, 227]. For a meaningful assessment, we develop a total of 81 ML-PWD, each trained over a different dataset (among the three we consider), using a different learning algorithm (among 9 well-known classifiers), and analyzing a specific set of features (based on either the URL, the HTML, or both). Let us explain all these procedures, most of which leverage the open-source implementation of SpacePhish [59] (which received a reusable artifact at ACSAC'22, and replicated in [190]).

Feature sets. Considering the difference between Chinese and Western websites (§5.2), we generate the feature representation of each website by extracting a total of 65 features (i.e., F), shown in Table 5.3. Of these, 60 are computed by leveraging the open-source feature extractor also released with SpacePhish [59] (we stress that this tool is based on the well-known work by Mohammad et al. [188]—which also overlaps with [184, 261]), which we enhance with some *original ideas* (inspired by [130, 139, 218]); whereas 5 (in boldface) are specific to Chinese websites: to extract them, we propose our *own implementation* of some recommendations by [270] (one of the few papers on Chinese PWD). (More details are in Appendix 5.9.) Overall, our 65 features can be divided into three sets:

- F_u, i.e., 36 URL-based features used in existing PWD, which mainly relate to domain and path.
- F_h, i.e., 29 features extracted from HTML contents (e.g., information of links, iframe, form, button, etc.);
- F_c, i.e., the combination of all features in Table 5.3.

We consider these three perspectives for both a "research and practical" reason. First, because it allows one to conduct an *ablation study* (we will do this in §5.6.1). Second, because some ML-PWD may not analyse the URL, whereas others may not analyse the HTML (this can be done to make the analysis faster, or to enable PWD when some information is missing, or even to create "adversarially robust" ML-PWD, according to [59]).

Learning algorithm. We consider 9 ML algorithms that support binary classification—all of which have been used in previous ML-PWDs [59, 65, 97, 137, 147, 222, 227, 247]. Specifically, 7 are "shallow" ML algorithms: Random Forest (RF), Logistic Regression (LR), Decision Tree (DT), Gradient Boosting (GB), AdaBoost (AB), Support Vector Machines (SVM) and K-Nearest Neighbors (KNN); while 2 are deep learning algorithms: Multi-Layer Perception (MLP) and Convolutional Neural Network (CNN).

Datasets and Setup. The evaluation of our ML-PWD entails our three 'language' datasets (§5.3): `ChiPhish`, `EngPhish`, `WstPhish`. For every sample (i.e., a website) in each dataset, we generate three variants by extracting a given feature representation—each corresponding to a specific feature set among the three described earlier (F_u, F_h, F_c). This procedure yields 9 different sets of samples (i.e., 3 feature sets $\times$ 3 language datasets). Each of these sets is then divided into train:test partitions with an 80:20 split (common in ML-PWD [48, 59, 72]). To account for randomness in the split and reduce the chances of biased results, we repeat all our experiments 10 times—thereby allowing one to derive statistically significant conclusions. These experiments are done on Ubuntu 20.04 system with CPU Intel Xeon W-2223 @ 3.60GHz; we report the training and testing runtime in Table 5.10 (for the CNN, we do not use GPU acceleration for a fair comparison). We release our (documented) source code for reproducibility [2].

TABLE 5.3: The features considered in our evaluation. Features in boldface are specific of Chinese websites. Features whose name starts with $U\_$ denote F_u, and those starting with $H\_$ denote F_h; finally, F_c comprises all features in the table.

#	Feature Name	#	Feature Name	#	Feature Name
1	U_dash	23	U_ip	45	H_rClick
2	U_tldinSub	24	U_at	46	H_brokenLin
3	U_pageRank	25	U_pt	47	H_loginForm
4	U_ssl	26	U_unicode	48	H_hidDiv
5	U_abn	27	U_age	49	H_statBarMod
6	U_numerical	28	U_rdr	50	H_css
7	U_tldinPath	29	U_dns	51	H_anchors
8	U_shortestWrdPath	30	U_tldNum	52	H_commRatioFt
9	U_lngHost	31	U_punycode	53	H_DominCopr
10	U_regLen	32	U_lngWrdPath	54	H_hidInp
11	U_senwrd	33	U_avgHost	55	H_iframe
12	U_totwrdUrl	34	U_avgWrdPath	56	H_favicon
13	U_shortestWrdUrl	35	U_SER	57	H_exItem
14	U_shortestWrdHost	36	U_GI	58	**H_icpCode**
15	U_lngWrdUrl	37	H_SFH	59	**H_ecert**
16	U_avgWrdUrl	38	H_popUp	60	H_freqDom
17	U_statsRep	39	H_nulItem	61	H_obj
18	U_len	40	H_metaScrpLin	62	H_commPage
19	U_shorter	41	**H_icpReg**	63	H_nulLin
20	U_sub	42	**H_icpDom**	64	H_nulLinFt
21	U_commItemNum	43	**H_icpApp**	65	H_hidBtn
22	U_pathExtend	44	H_titBr		

What about image-based PWD? We do not consider these PWD for our RQ because they are *demonstrably inappropriate*. We provide factual evidence (theoretical and empirical) in Appendix 5.12, where we also showcase a failed experiment.

5.4.2 Closed-source Phishing Website Detectors

For our last RQ we must assess the effectiveness of real PWD on Chinese and Western websites. Such PWD are closed-source since they are developed by security companies, and can leverage either signature- or ML-based detection techniques. To provide a meaningful answer to RQ3, we consider both *production-grade* PWD services (§5.4.2) as well as *competition-grade* PWD using machine learning (§5.4.2), which are well-known in the "West".

Production-grade PWD

There exist several commercial tools that can be used to determine whether a website is malicious. These tools accept as input either the URL or the HTML of a website, and then analyze such input in a black-box manner; these tools can rely on up-to-date blocklists, but they may also query third-party services that perform a deeper analysis. Notable examples of such services are: Virustotal [42], Netcraft [35], or PhishDetector [36]. For our evaluation we rely on **VirusTotal**, since its output accounts for the responses of dozens of scanners and URL/Domain blocklists (in contrast, Netcraft and PhishDetector only consider the response of a single tool), and is widely used by the research community [52, 83, 93, 102, 212]. Specifically, we submit the HTML of our samples (benign and phishing) to VirusTotal: every "query" to VirusTotal corresponds to having 61[13] PWD (each leveraging proprietary detection methods) to analyse the corresponding sample – allowing us to provide a broad perspective on the detection capabilities of real systems. Furthermore, we will also consider **Google Safe's Browsing** (GSB), which allegedly also uses ML [30].

Competition-grade ML-PWDs

To provide a complementary perspective to our custom-developed ML-PWD (§5.4.1) we find it instructive to also consider the analysis provided by a (closed source) ML-PWD that is known to use ML techniques. Specifically, we consider the anti-phishing detectors provided for the well-known Machine Learning Security Evasion Competition (MLSEC) organized by CujoAI in 2022 [99]. These ML-PWD (8 in total) analyze the raw HTML of a webpage as input, and provide a 'phishing' confidence (within the [0–1] range, with 0 denoting a benign sample and 1 a phishing sample) as output. The organizers of MLSEC allowed the research community to use their ML-PWD for three months after the challenge ended in September 2022. We took this opportunity to test these detectors on the raw HTML of every webpage in our three datasets—thereby ensuring a consistent setup as the one in §5.4.2 (for VirusTotal).

5.4.3 Summary and Workflow

We summarize how we combined all the elements discussed insofar to answer our three RQ (see Fig. 5.7).

[13] We perform our analysis in Dec. 2022, but this number can change [257]. At that point in time, the detectors queried by VirusTotal are 78, but 17 of these returned an error so we will not consider them.

i) We create three 'language' datasets. We collect a new dataset of Chinese-only websites (`ChiPhish`); and we extract subsets of English-only (`EngPhish`) and generic Western (`WstPhish`) websites by drawing samples from existing datasets ([65, 252]) by means of our Language Selector Tool (LaSeTo).

ii) We develop a feature extractor and use it to compute 65 features for each sample in our three 'language' datasets; in such a way, every sample can be seen as three different variants—depending on which features are used to represent it (F_u, F_h, or F_c).

iii) Altogether, these 3 feature sets and 3 'language' datasets yield the 9 datasets to assess state-of-the-art ML-PWD, which leverage one among 9 ML algorithms. We train classifiers on 80% of each of our 9 datasets, thereby leading to 81 ML-PWD.

iv) Then, (for RQ1 and RQ2) we test each of our 81 ML-PWD on the remaining 20% of each dataset[14] (of either the same or different 'language'). Finally, for RQ3, we submit the raw HTML of every sample in each of our 'language' datasets to the considered real (and closed-source) PWD (both the production- and competition-grade PWD).

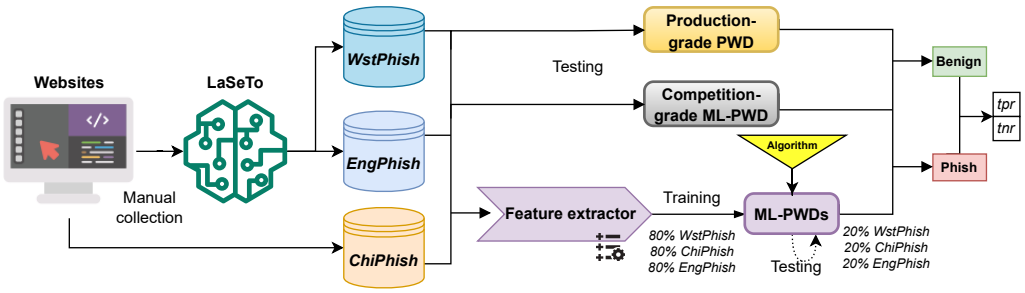
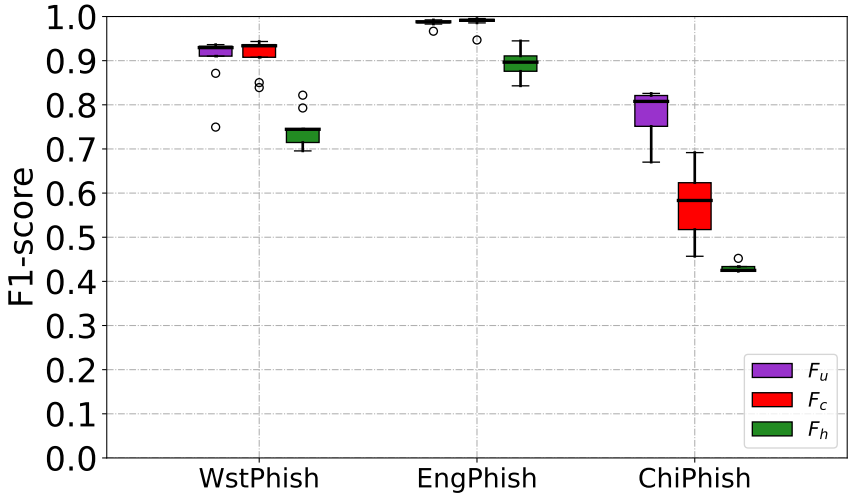


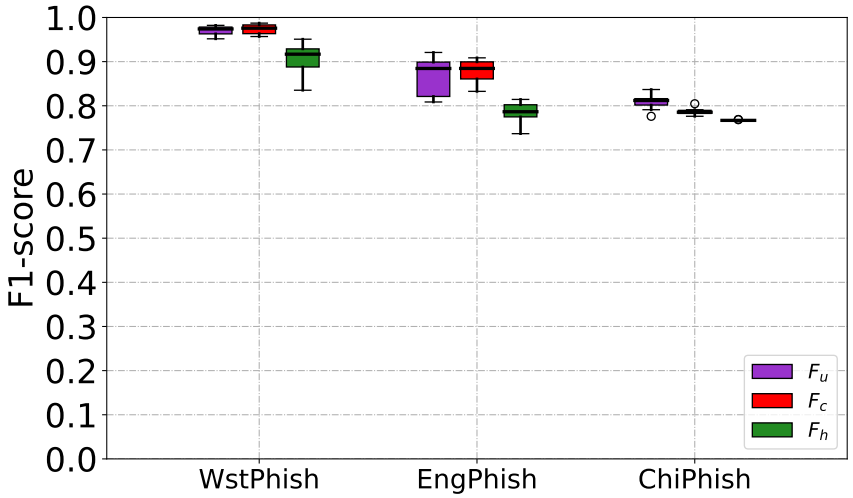
FIGURE 5.7: Overview of our evaluation workflow.

We measure the performance of each PWD by computing the true positive rate (tpr), true negative rate (tnr) and F1-score ($F1$), which is customary in our context [142]. It is desirable that a PWD exhibits high tpr and high tnr (i.e., high detection rate with low false positive rate).

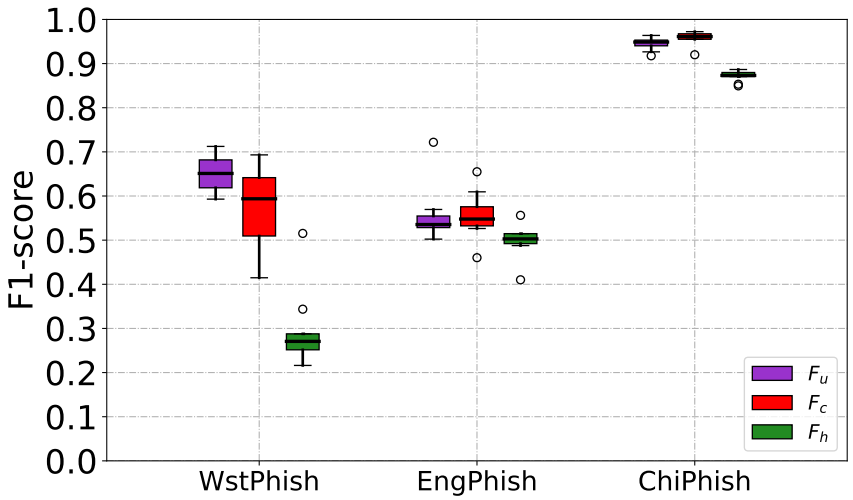
[14] Such testing is done by considering a 'matching' feature set: a classifier analysing F_u at training-time will also analyse F_u at inference.



(A) Test partition: `EngPhish`.



(B) Test partition: `WstPhish`.



(C) Test partition: `ChiPhish`.

FIGURE 5.8: Cross-language performance of state-of-the-art ML-PWD. The graphs show the distribution of the $F1$-score (y-axis) of our ML-PWD, trained on a specific dataset (x-axis) and analyzing a given feature set (legend), on the test partition of each language dataset (subfigure). The bins aggregate the results of all our considered learning algorithms across the 10 trials.

5.5 Main Results and Answers

We focus our attention to answering our three research questions, which we tackle by performing a large experimental campaign using our datasets (described in §5.3) to assess the capabilities of various PWD (described in §5.4).

We first consider the experiments on the open-source ML-PWD for RQ1 and RQ2 (§5.5.1), and then consider those entailing the real PWD for RQ3 (§5.5.2). We also extend our findings by assessing a production-grade PWD developed by Chinese companies (§5.5.3).

5.5.1 Assessment of research ML-PWD (RQ1,2)

To simultaneously answer both RQ1 and RQ2, we perform a "cross-language" evaluation of our state-of-the-art ML-PWD. These experiments will also allow one to assess whether ML-PWD trained on generic Western websites (i.e., using diverse phonological languages) work well on English-only websites (see §5.3.1).

Procedure

We adhere to the workflow in §5.4.3. For each 'language' dataset (i.e., `ChiPhish`, `WstPhish`, `EngPhish`) we create three different variants depending on a given feature set (i.e., F_u, F_h, F_c), yielding 9 evaluation sets. Then, we use 80% of each of these sets to train 9 different classifiers (by changing the learning algorithm), for a total of 81 ML-PWD. Finally, we use the remaining 20% of each set to measure the performance of each ML-PWD that analyses the same features. We repeat all of these experiments 10 times.[15] We report the complete results of this evaluation in Appendix 5.10.1, showing the *tpr* and *tnr* of all our 81 ML-PWD. In what follows, we will provide a high-level analysis and then focus on the performance of the best ML-PWD.

High-level analysis

We provide in Figs. 5.8 a comprehensive overview of our results. These figures report the F1-score (aggregated across the 10 trials and 9 learning algorithms) achieved by our ML-PWD for each different test set. For example, Fig. 5.8c shows the distribution of the $F1$-score for the ML-PWD (which vary either for their training dataset or feature set) when tested on `ChiPhish` (assuming a matching feature set). From Figs. 5.8, we can see that our ML-PWD, when tested on websites having the same language as their training data, work well—and this is especially the case for ML-PWD using F_c, which always outperform those analysing fewer features. We also appreciate that the ML-PWD trained on either `EngPhish` or `WstPhish` exhibit similar performance when tested on (respectively) `WstPhish` or `EngPhish`. However, the situation changes when our ML-PWD on phonological (resp. hieroglyphical) languages must analyse hieroglyphical (resp. phonological) languages. This is evident when testing on samples from `ChiPhish` (Fig. 5.8c): despite the good performance of the "Chinese" ML-PWD (the $F1$ is almost always$>$0.85), the "Western" and "English" detectors have a significant drop (almost never above 0.70 $F1$), which is inappropriate to analyse Chinese websites. Similarly, by observing Fig. 5.8a, we can see that the "Chinese" ML-PWD work poorly on English websites; interestingly, however, they

[15] Note that, to follow best-practices [62, 70] and ensure consistency, we use the *same* test-set to each ML-PWD for each trial. E.g., we use the same 20% of `ChiPhish` to test all the ML-PWD, and then start a new trial by randomly sampling a new training and test partitions—which we use to develop and assess new ML-PWD.

still retain around 0.75 $F1$ on the generic Western websites in `WstPhish`. Even though this result appears encouraging, the complete results in Appendix 5.10.1 reveal that such "encouraging" $F1$ conceals an *unacceptable rate of false positives*. E.g., for the ML-PWD analysing F_c (see Table 5.11), $tnr \in [0.08, 0.27]$, making these ML-PWD clearly unusable in practice.

Best ML-PWD

We now focus on the best ML-PWD of our evaluation, i.e., the one using a specific learning algorithm (out of 9) to analyze a given feature set (out of 3) that achieved the highest $F1$ as measured on the test partition of the corresponding training dataset. This is because only the best ML-PWD would be (hypothetically) deployed in reality, and hence its results are more appropriate to derive sensible conclusions. The ML-PWD analyzing F_c and using RF consistently outperformed the others (a result which aligns with prior work, e.g., [59]). Hence, we provide the results of these ML-PWD in Table 5.4, showing the average $F1$ (and std.dev.) across the 10 trials. We make the following observations.

- *Same language.* The $F1$ are highest on the diagonal (0.97 and 0.99), indicating that our ML-PWD can correctly classify websites of the same language (results aligning with [59] for `WstPhish`, and with [65] for `EngPhish`); we also find it positive that our `ChiPhish` yields effective ML-PWD for Chinese websites.
- *Chinese vs Phonological languages.* There is a remarkable performance drop when the ML-PWD trained on either `WstPhish` or `EngPhish` is tested on `ChiPhish`, exhibiting an $F1$ of only 0.52 and 0.55. Plus, the ML-PWD trained on `ChiPhish` is less effective while predicting websites in phonological languages, with an $F1$ of 0.78 on `WstPhish` and of 0.58 on `EngPhish`.
- *Generic vs English-only languages.* Interestingly, the ML-PWD trained on `WstPhish` can correctly predict the websites in `EngPhish` ($F1$=0.94), whereas the ML-PWD trained on `EngPhish` is only slightly inferior when tested on `WstPhish` ($F1$=0.91).

TABLE 5.4: Cross-language performance of the best ML-PWD (RF and F_c) on our three datasets. Cells report the avg $F1$ (and std) over 10 trials. See Table 5.11 for the complete tpr and tnr.

F1-score Test (20%) / Train (80%)	`ChiPhish`	`WstPhish`	`EngPhish`
`ChiPhish`	0.97±0.007	0.78±0.007	0.58±0.075
`WstPhish`	0.52±0.044	0.99±0.003	0.94±0.006
`EngPhish`	0.55±0.023	0.91±0.006	0.99±0.001

ANSWER TO RQ1 AND RQ2: State-of-the-art ML-PWD tailored for Chinese websites work poorly[a] on "Western" websites, and vice-versa.

[a] This finding goes against the one in [273].

We provide in Appendix 5.13 the results we obtained by considering the vanilla ML-PWD of SpacePhish [59].

5.5.2 Assessment of real PWD systems (RQ3)

We assess closed-source PWD developed by practitioners.

VirusTotal

We submit the raw HTML of every sample in each of our three datasets (`ChiPhish`, `EngPhish` and `WstPhish`) to VirusTotal, which automatically forwards it to 61 cyber detectors (provided by security companies) and then reports the output of each of these. Such an output can come in various forms: we consider the responses {malicious, malware, phishing, suspicious} as a "phishing" prediction; whereas {clean, undetected} correspond to a "benign" prediction. Hence, we calculate the tpr and fpr of each detector by comparing its prediction with the respective ground truth. The **results** are reported in Table 5.5, from which two intriguing findings emerge.

- These detectors are fine-tuned to *minimize false positives*: all detectors achieve a perfect tnr on our data.
- These detectors perform *terribly* in identifying the phishing samples in `ChiPhish`: the best detector, *AVG*, has a tpr of 0.03; in contrast, it can detect phishing webpages in `WstPhish` and `EngPhish` much better, with a tpr of 0.49 and 0.52 respectively.

While it is understandable that operational detectors minimize false alarms, it is concerning that *none of these real systems can detect Chinese phishing* (the avg tpr across 61 detectors is 0.004). Notably, these detectors (aside from *AVG* and *Avast*) work poorly also on `WstPhish`.

Google Safe Browsing

We assess the capabilities of GSB. We submit all the samples in our three datasets to the GSB API [32] (which accepts URL as inputs) and we record how many webpages trigger a "suspicious" response. The results are as follows:

- `EngPhish`: tpr=0.043 (only 176 phishing samples are detected), with no false positives (i.e., tnr=1.0).
- `WstPhish`: tpr=0.004 (only 26 phishing samples are detected), with only one false negative (tnr=0.999).
- `ChiPhish`: tpr=0.002 (only 1 phishing sample is detected) with no false positives (tnr=1.0).

From these results, it is apparent that even this very important anti-phishing tool, deployed in popular web-browsers, is unable to identify Chinese phishing websites.

Competition-grade ML-PWD (MLSEC)

We submit the raw HTML of all samples in our three datasets to each of the 8 "black-box" ML-PWD of MLSEC. The output of each of these detectors is a confidence score (from 0.0 to 1.0): for this experiment, we consider a score that is higher than 0.5 to denote a "phishing" prediction (and "benign" otherwise). We then check these predictions against the corresponding ground-truth, and derive the tpr and tnr. The **results** are in Fig. 5.9, showing the distribution of the tpr and tnr across the 8

TABLE 5.5: Performance in VirusTotal, reported as the *tpr* and *tnr*.

Anti-phishing Service	WstPhish		EngPhish		ChiPhish		Anti-phishing Service	WstPhish		EngPhish		ChiPhish	
	tpr	*tnr*	*tpr*	*tnr*	*tpr*	*tnr*		*tpr*	*tnr*	*tpr*	*tnr*	*tpr*	*tnr*
Bkav	0.00	1.00	0.00	1.00	0.00	1.00	*Lionic*	0.01	1.00	0.08	1.00	0.00	1.00
MicroWorld	0.05	1.00	0.20	1.00	0.01	1.00	*Panda*	0.00	1.00	0.00	1.00	0.00	1.00
ClamAV	0.00	1.00	0.00	1.00	0.00	1.00	*CMC*	0.00	1.00	0.00	1.00	0.00	1.00
QuickHeal	0.09	1.00	0.12	1.00	0.01	1.00	*McAfee*	0.02	1.00	0.04	1.00	0.00	1.00
Malwarebytes	0.00	1.00	0.00	1.00	0.00	1.00	*Zillya*	0.01	1.00	0.03	1.00	0.00	1.00
K7AntiVirus	0.00	1.00	0.00	1.00	0.00	1.00	*AVG*	**0.49**	**1.00**	**0.52**	**1.00**	**0.03**	**1.00**
K7GW	0.00	1.00	0.00	1.00	0.00	1.00	*NOD32*	0.14	1.00	0.20	1.00	0.00	1.00
Baidu	0.01	1.00	0.04	1.00	0.00	1.00	*VirIT*	0.00	1.00	0.01	1.00	0.01	0.99
Cyren	0.05	1.00	0.17	1.00	0.00	1.00	*Fortinet*	0.11	1.00	0.22	1.00	0.00	1.00
Symantec	0.01	1.00	0.11	1.00	0.00	1.00	*AhnLab*	0.01	1.00	0.04	1.00	0.00	1.00
HouseCall	0.00	1.00	0.04	1.00	0.00	1.00	*Avast*	0.49	1.00	0.52	1.00	0.03	1.00
Cynet	0.08	1.00	0.18	1.00	0.01	1.00	*Kaspersky*	0.01	1.00	0.04	1.00	0.00	1.00
BitDefender	0.05	1.00	0.19	1.00	0.01	1.00	*NANO*	0.02	1.00	0.20	1.00	0.01	1.00
SuperAntiSpyw.	0.00	1.00	0.00	1.00	0.00	1.00	*Tencent*	0.01	1.00	0.06	1.00	0.00	1.00
Ad-Aware	0.05	1.00	0.17	1.00	0.01	1.00	*Sophos*	0.00	1.00	0.06	1.00	0.00	1.00
Comodo	0.01	1.00	0.16	1.00	0.00	1.00	*F-Secure*	0.01	1.00	0.02	1.00	0.00	1.00
DrWeb	0.00	1.00	0.05	1.00	0.00	1.00	*VIPRE*	0.03	1.00	0.15	1.00	0.01	1.00
TrendMicro	0.00	1.00	0.04	1.00	0.00	1.00	*McAfee-GW*	0.02	1.00	0.07	1.00	0.01	1.00
FireEye	0.05	1.00	0.20	1.00	0.01	1.00	*Emsisoft*	0.05	1.00	0.17	1.00	0.01	1.00
Ikarus	0.09	1.00	0.23	1.00	0.03	1.00	*GData*	0.07	1.00	0.22	1.00	0.01	1.00
Jiangmin	0.00	1.00	0.00	0.99	0.00	1.00	*ZoneAlarm*	0.01	1.00	0.03	1.00	0.00	1.00
Avira	0.08	1.00	0.18	1.00	0.01	1.00	*Antiy-AVL*	0.02	1.00	0.12	1.00	0.00	1.00
Kingsoft	0.00	1.00	0.00	1.00	0.00	1.00	*Gridinsoft*	0.00	1.00	0.00	1.00	0.00	1.00
Arcabit	0.04	1.00	0.15	1.00	0.01	1.00	*ViRobot*	0.00	1.00	0.00	1.00	0.00	1.00
Microsoft	0.02	1.00	0.17	0.99	0.00	1.00	*Google*	0.11	1.00	0.28	1.00	0.02	1.00
Acronis	0.00	1.00	0.00	1.00	0.00	1.00	*VBA32*	0.00	1.00	0.00	1.00	0.00	1.00
ALYac	0.04	1.00	0.15	1.00	0.01	1.00	*MAX*	0.05	1.00	0.19	1.00	0.01	1.00
Zoner	0.00	1.00	0.01	1.00	0.01	1.00	*BitDefenderΘ*	0.00	1.00	0.00	1.00	0.00	1.00
Rising	0.01	1.00	0.05	1.00	0.00	1.00	*Yandex*	0.00	1.00	0.00	1.00	0.00	1.00
Tachyon	0.00	1.00	0.00	1.00	0.00	1.00	*MaxSecure*	0.01	1.00	0.04	1.00	0.00	1.00
Sangfor	0.04	1.00	0.17	1.00	0.00	1.00	AVERAGE	0.04	1.00	0.11	1.00	0.004	1.00

MLSEC detectors for each of our three datasets (the detailed performance of each detector is in Table 5.14). From Fig. 5.9 we see that these detectors perform much better on WstPhish (avg *tpr*=0.60) and EngPhish (avg *tpr*=0.64) compared to ChiPhish (avg *tpr*=0.27). Even though these detectors are for competitions[16], these results further show that *Chinese websites are rarely accounted for* when designing ML-PWD.

ANSWER TO RQ3: PWD developed by security practitioners can hardly identify Chinese phishing webpages. In contrast, these PWD are more effective against phishing webpages in phonological "Western" languages.

5.5.3 Production-grade Chinese PWD

Insofar, we only considered real PWD (allegedly) tailored for Western websites, i.e., we did not evaluate if PWD used by Chinese companies work well on websites in phonological languages. This is because all such "production-grade Chinese PWD" *are not readily accessible by researchers*. For instance, VirusTotal provides APIs that can be used to automate the querying process, thereby enabling the analyses of thousands of websites in a humanly feasible way. Unfortunately, we are not aware of similar "Chinese-specific" services, since they all require each request to be manually submitted, thereby preventing the analysis of all samples in our datasets.

[16]The organizers of MLSEC admittedly tweaked the detectors so that they would be harder to evade (explaining the underwhelming *tnr*).

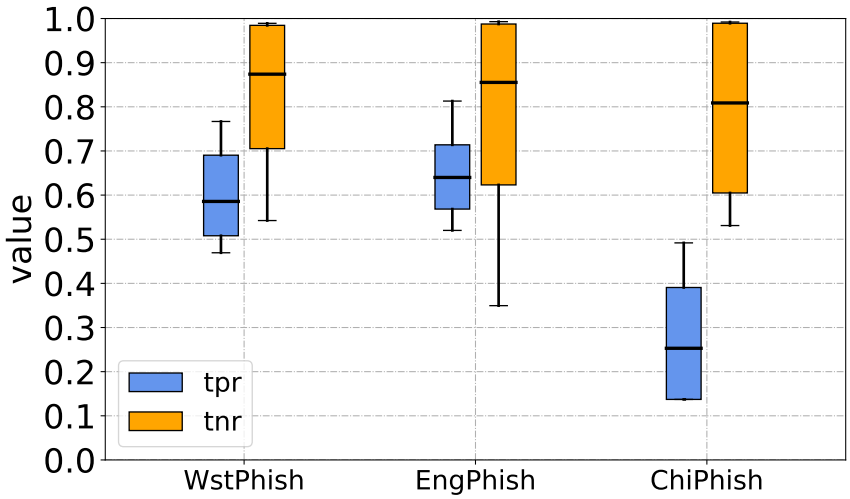


FIGURE 5.9: Performance of the ML-PWD provided by MLSEC on `WstPhish`, `EngPhish` and `ChiPhish`, reported as *tpr* and *tnr*.

Nonetheless, to inspire future research and provide further evidence of the existence of a gap, we performed a proof-of-concept experiment in which we submit 200 "Western" phishing samples to a commercial Chinese PWD. Specifically, we consider VenusEye [41], one of the most popular Chinese tools for threat intelligence. VenusEye detects phishing by checking a given URL against closed-source blocklist. We randomly sample 100 phishing samples from both `WstPhish` and `EngPhish`, and submit the corresponding URL to VenusEye (in March 2023). Accordingly, 74 and 82 of the samples from `WstPhish` and `EngPhish` are flagged as phishing. To demonstrate "the gap", we tested these very same 200 samples on the URL version of VirusTotal, which achieved a 97% *tpr*.

Takeaway. Chinese production-grade PWD (using blocklists) cannot detect 20% of our submitted Western phishing samples.

5.6 Bridging the Gap (analysis and solutions)

We enrich our contribution by providing some critical analyses and additional experiments. Specifically, we attempt to explain our results (§5.6.1); then, we propose and practically evaluate two potential solutions (§5.6.2 and §5.6.3) to bridge the gap between Chinese and Western PWD .

5.6.1 Explanations (ablation study)

Intuition. The reason why Chinese (resp. Western) PWD work poorly on Western (resp. Chinese) websites can be traced back to the semantic difference between Chinese and Western websites (which we presented in §5.2). Such a difference leads to samples having a different feature distribution, thereby preventing a correct analysis by any ML-PWD that is trained and tested on websites from different "regions". To identify potential mitigations to the problem elucidated by our paper, we now try to explain our results by focusing the attention on the features analysed by our[17] ML-PWD.

[17]The "black-box" nature of the PWD considered by VirusTotal, GSB, and MLSEC prevent one to perform sensible a-posteriori analyses.

Feature Importance. Investigating the most relevant features for classification is a well-known technique for studying the underlying logic learned by an ML model [63]. To align such an analysis to §5.5.1, we report in Fig. 5.10 the ranking (as given by scikit-learn) of the top10 features for the best ML-PWD, i.e., RF analysing F_c. (We provide an analysis of the feature ranking for the RF analysing F_u and F_h in Appendix 5.11.) From Fig. 5.10, we see that two Chinese-specific features, 'H_icpApp' and 'H_icpCode', appear in the top10 of `ChiPhish`, and that 'H_icpApp' is the second most important feature; in contrast, neither of these features are relevant for the classifiers trained on `WstPhish` and `EngPhish`. This situation can explain why classifiers trained in `ChiPhish` work poorly on Western websites (i.e., `WstPhish` and `EngPhish`). At the same time, by observing the rankings for the classifiers trained on `WstPhish` and `EngPhish`, we see that both have 8 features in the top-10: this suggests why classifiers trained on `WstPhish` and `EngPhish` perform similarly. Moreover, both 'H_icpApp' and 'H_icpCode' are the features extracted from the HTML, which can verify our intuition that the gap between Western and Chinese PWD is more manifested in HTML.

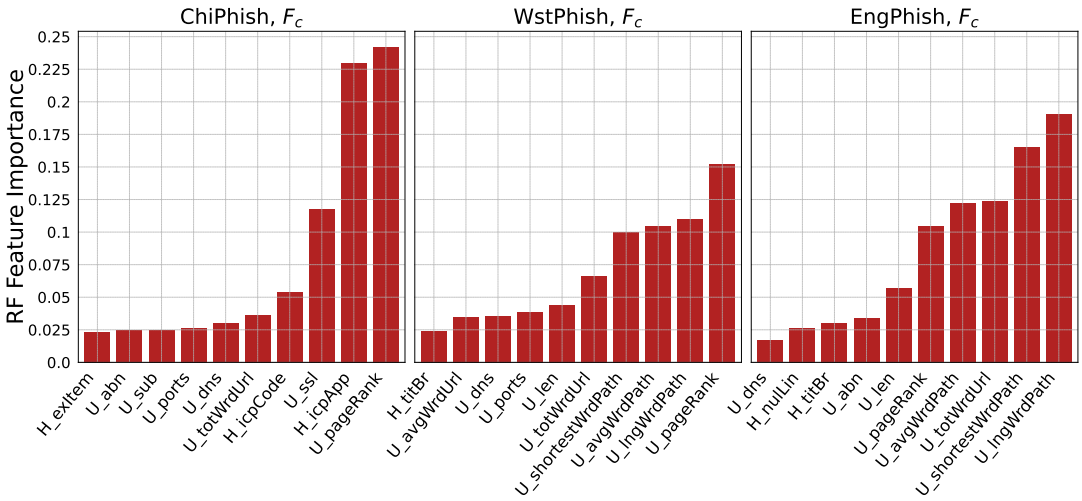


FIGURE 5.10: Top 10 features of RF (F_c) trained on each dataset.

Mitigation. A possible way to reduce the performance gap between our Chinese ML-PWD and the ML-PWD focusing on phonological languages is by considering a feature set that is "less specific" to Chinese websites. This could be done by developing ML-PWD that analyse *only the URL*, i.e., F_u. In fact, by looking at the boxplots in Figs. 5.8, we can see that the ML-PWD analysing F_u (purple bins) tend to have a higher $F1$ than those using F_h and F_c trained on Chinese (resp. non-Chinese) and tested on non-Chinese (resp. Chinese) websites. For instance, the ML-PWD trained on `ChiPhish` using F_u obtain nearly 0.8 $F1$ (up from 0.6 of F_c) when analysing `EngPhish` (see Fig. 5.8a); whereas the ML-PWD trained on `WstPhish` obtain 0.65 $F1$ (up from 0.60 of F_c) when tested on `ChiPhish` (see Fig. 5.8c). However, in both cases, such gain comes at the expense of a reduced $F1$ when analysing websites of the same language: classifiers trained on F_c are *always* statistically superior than those using F_u on the respective language dataset (a student t-test reveals this hypothesis to be true: $p<0.05$).

Takeaway. Using ML-PWD that analyse only the URL can work, but presents tradeoffs on websites of the same-language.

Finally, we refer the reader to Appendix 5.13, wherein we discuss the cross-language results we achieved by using the nine vanilla ML-PWD of SpacePhish on our datasets.

5.6.2 Towards an Universal ML-PWD

The most straightforward way to "bridge the gap" between Western and Eastern PWD is to create an universal dataset containing samples from various languages. Insofar, we have considered ML-PWD trained on websites of a specific language group. Hence, we now merge all our three datasets and scrutinize the effectiveness of an ML-PWD trained (and tested) on the resulting "universal" dataset.

Method. We merge `ChiPhish`, `WstPhish`, `EngPhish`, and then split the resulting dataset into train:test partitions (using the same 80:20 ratio as we did in §5.4.1). We then extract the F_c feature representation of each sample, and use the corresponding training partition to develop a ML-PWD using RF as classification algorithm (because it outperformed all other algorithms, see Table 5.11). We use the test partition to measure its performance (*tpr*, *tnr*, *F*1); we repeat this 10 times to reduce bias. We report the results in Table 5.6, where rows denote a given metric, and columns denote a specific subset of the test partition.

TABLE 5.6: **Universal ML-PWD**: we train and test an RF (F_c) on all our datasets (using an 80:20 split), and we measure the performance (avg and std.dev) on each language dataset.

	`ChiPhish`	`WstPhish`	`EngPhish`
tpr	$0.89_{\pm 0.023}$	$0.99_{\pm 0.002}$	$0.97_{\pm 0.004}$
tnr	$0.99_{\pm 0.004}$	$0.95_{\pm 0.006}$	$0.99_{\pm 0.001}$
*F*1	$0.94_{\pm 0.001}$	$0.98_{\pm 0.002}$	$0.98_{\pm 0.004}$

Considerations. In general, we can see that our universal ML-PWD works well: the *F*1 is always above 0.94, and the worst false positive rate (i.e., 1-*tnr*) is of only 0.05. However, by comparing these results with those in Table 5.4, we can see that our universal ML-PWD has a slightly inferior *F*1 (∼0.01) on a specific language group w.r.t. the ML-PWD specifically trained on that very same group. Furthermore, by comparing Table 5.6 with the detailed results in Table 5.11 (for RF), we see that the drop is significant in two cases: the *tpr* on `ChiPhish` (which drops from 0.96 to 0.89), and the false positive rate on `WstPhish` (which increases from 0.025 to 0.05).

Takeaway. Mixing datasets of Chinese and Western languages improves the *tpr*, but can double the false positive rate. Such is the "price to pay" for deploying our universal ML-PWD.

5.6.3 Exploiting our LaSeTo: a novel ML-PWD

As a final potential mitigation, we propose an intuitive solution rooted in our self-developed LaSeTo (§5.3.3). We are inspired by the remarkable results achieved by our "language-specific" ML-PWD (see §5.5.1): indeed, our ML-PWD work well *if they analyse websites in a language they "have seen"*. We use this observation as a scaffold and develop an *original phishing website detection system*[18] which integrates an "ensemble" of our custom ML-PWD which are put in a pipeline to our self-developed LaSeTo. A schematic of this system is in Fig. 5.11.

Evaluation. We partition each of our three datasets in train:test by following the usual 80:20 split. We train three "language-specific" ML-PWD (one per dataset) on

[18] We are not aware of existing anti-phishing schemes that entail a "language selector" before the detection model (as we also stated in §5.2).

80% of each language dataset (we use F_c and RF). Then, we merge the remaining 20% of each dataset into a single Test Dataset. Next, we use LaSeTo to analyse the language of each sample in this Test Dataset: if the language is Chinese, the sample is analysed by the ML-PWD trained on `ChiPhish`; if the language is English, it is analysed by the ML-PWD trained on `EngPhish`; otherwise, it is analysed by the ML-PWD trained on `WstPhish`. We repeat this process 10 times, and report the results in Table 5.7.

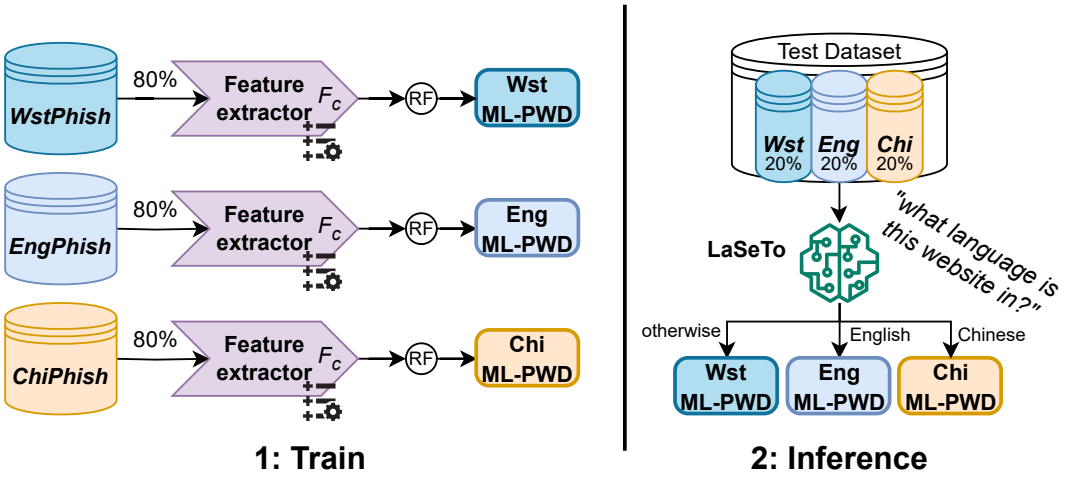


FIGURE 5.11: Proposed phishing website detection system. We train language-specific ML-PWD (left), and then use our self-developed LaSeTo to determine the language of any given webpage, which is forwarded to the most suitable ML-PWD (right).

Analysis and Feasibility. At a high-level, this solution represents a viable alternative to the "universal" ML-PWD (§5.6.2). Indeed, by using LaSeTo it is possible to develop a system that works much better on "Western" websites, while still achieving a satisfactory performance on Chinese websites—albeit slightly inferior (e.g., 0.88 $F1$ vs 0.94). We also stress that these results are slightly inferior than the "baseline" ones (shown in Table 5.4 and Table 5.11) because LaSeTo presents a small margin of error (as we measured in §5.3.3), which may, e.g., lead it to forward a Chinese website to the incorrect model. However, by improving LaSeTo, it would be possible to approximate the near-perfect performance of the language-specific ML-PWD on their respective datasets. Finally we note that operational PWD must be fast at processing a webpage [107, 164]. Hence, to demonstrate the feasibility of our solution, we have measured the *runtime* for using LaSeTo: on average, it requires 0.04s to output the language of a given webpage[19]. Such a low overhead makes our tool appropriate for real-time analyses (we also measured the runtime for all our ML-PWD in Table 5.10).

Takeaway. Using a custom tool to infer the language of a given webpage, and then use its output to select the appropriate detection technique is helpful for "multi-language" PWD.

[19]We measured this on commodity hardware and after recording the time required to process all samples in our three datasets.

TABLE 5.7: **ML-PWD integrating LaSeTo**: we report the performance (avg and std.dev., computed over 10 trials). Overall (on a generic webpage): tpr=0.98±0.0029, tnr=0.99±0.0022.

	ChiPhish	WstPhish	EngPhish
tpr	0.85±0.025	0.99±0.003	0.99±0.003
tnr	0.95±0.012	0.98±0.006	1.00±0.000
$F1$	0.88±0.017	0.99±0.002	0.99±0.001

5.7 Discussion and Future Work

Our evaluation revealed that the gap between Chinese and Western PWD *exists*, and it is significant since it affects both *research and practice*. Let us discuss some potential limitations of our study and identify room for future work.

Focus of this study. Given the scarce literature on this subject (see §5.1.2), our primary goal is to provide *factual evidence* (theoretical and empirical) that reveals the gap between Chinese and Western PWD. To this purpose, we scrutinize *existing* techniques (open source and proprietary) for PWD and highlight their limitations (further expanded in Appendix 5.13). As a constructive step forward, we propose mitigations and we release our code, tools, and data (including our proposed `ChiPhish` dataset, the first of its kind). We hope that our resources will spearhead the development of novel techniques that can bridge the gap we brought to light. For instance, future endeavours can use `ChiPhish` as a basis for an "universal" ML-PWD (which improves our attempt in §5.6.2), or enhance LaSeTo.

Size of our datasets. As shown in Table 5.2, the amount of data in our three language datasets varies considerably, with 15 111 samples for `EngPhish`; 11 204 for `WstPhish`; and 1 620 for `ChiPhish`. We acknowledge that such differences may hinder the generalizability of our findings. However, as shown in Table 5.4, the baseline performance (i.e., same-language) of our ML-PWD measured on samples from either `WstPhish` or `EngPhish` aligns with the one achieved by prior work (i.e., [59, 65]). Furthermore, the performance on `ChiPhish` is also high, suggesting that (despite the smaller size) detectors can generalize well on samples from the same population. Nevertheless, we endorse future work to build upon our resources and expand our findings by, e.g., using `ChiPhish` to create a larger (and up-to-date) dataset of Chinese websites.

Considered ML-PWD. Prior literature proposed many ML-based methods against phishing websites (§5.1). *Analysing all such methods is clearly unfeasible for any single publication*—especially given that most such methods are not publicly available. In our evaluation, we rely on the open-source implementation of a state-of-the-art ML-PWD (i.e., the one in [59]), which we enhance by integrating our own implementation of features suggested by prior work on Chinese PWD (i.e., [270]), and further extend by considering more ML algorithms. We acknowledge that there exist ancillary ML-based methods for PWD, such as those entailing image similarity (e.g., [44]). However, such techniques are affected by shortcomings: first, they can

be trivially evaded [55, 92, 164]; second, they operate on the assumption that the visual representation of a website never changes [55]—and are hence prone to "out of distribution" problems (of a greater magnitude than our considered ML-PWD, since some features remain stable even among websites of different languages). For our goal, a meaningful assessment would have required to collect diverse visual representations of the *same* webpage—which are hard to find in the Chinese landscape.[20] Regardless, our extensive experiments on *real systems* (§5.5.2 and §5.5.3) reveal that the gap exists *in practice*.

Extension to other regions. The gap highlighted in this paper pertains to Chinese w.r.t. Western languages—and respective regions. However, phishing websites are also a problem in other areas besides China, each with its own language and regulations. Recent reports show an increasing trend of phishing websites in countries such as India [29], Japan [25], and the Middle East [19]. Unfortunately, these regions are vastly underrepresented in the PWD context (e.g., [47, 254]. Moreover, from a generic phishing perspective, few researches (e.g., [234, 244]) attempt to analyse the differences between such "minorities" and "Western" countries. Given the huge migration waves that interest Western countries (e.g., from the Middle East [243], China, India [182] or Africa [231]), we hope our findings can inspire future efforts to scrutinize whether such issues also affect other geographical areas.

Motivational Experiment. We scrutinize the effectiveness of existing PWD on *Japanese* (JP) and *Korean* (KR) websites. We collected a small sample of 200 webpages—of which 109 are benign (50 for JP and 59 for KR, taken from [38]) and 91 are phishing (50 for JP and 41 for KR, gathered from [23, 26, 180]). We train our best ML-PWD (RF using F_c on `WstPhish`) and test it on them; we repeat this ten times. For JP: $F1$=0.82±0.024 (tpr=0.74; tnr=0.94); for KR: $F1$=0.93±0.01 (tpr=0.93; tnr=0.95). We also submit these to GSB (for which the tnr=1.0, while the tpr is 0.04 for JP, and 0.0 for KR) and to VirusTotal (for which the average tpr is 0.12 for both JP and KR, but the tnr is 0.67 for JP, and 0.84 for KR). These results suggest that **real PWD work very poorly also on websites of these languages**—albeit research-PWD are more promising. Our repository also includes these webpages [2].

5.8 Conclusion

This paper aims to reveal and assess (and mitigate) the performance gap between Phishing Website Detectors (PWD) that analyse either Chinese or Western websites. We explain the *theoretical* differences that exist between phishing websites in China w.r.t. the Western side of the World, suggesting that PWD may behave differently if such differences are not accounted for. Then, after collecting the *first* dataset for Chinese-focused PWD, we *practically* demonstrate the existence of such a gap in modern PWD. We assess the performance of state-of-the-art PWD, spanning across: 81 variants of machine learning (ML) detectors proposed in research; 62 operational security services; and 8 competition-grade ML-PWD. Our large evaluation reveals that real systems (tuned to minimize false positives) can detect at best 3% of phishing Chinese websites—whereas they can detect around 50% for Western languages. Such an imbalance also affects ML-PWD, which exhibit high rates of false positives (sometimes above 70%) when assessed in a cross-language setting. Finally, we propose and investigate potential *fixes* (i.e., analysing diverse features, or mixing datasets), but all these attempts have a practical tradeoff (e.g., higher fpr).

[20] In App. 5.12, we show (via a **case-study** and **original experiments**) the limitations of *target-dependent* [97] PWD for Chinese websites.

TAKEAWAY. Existing PWD "in the West" are poorly equipped to counter phishing websites "in China" (and vice-versa). This is not acceptable given the constant migratory waves from/to these two sides of the World.

Our paper casts light on a hidden problem that likely also exists for other languages beyond Chinese. We encourage future efforts to **build upon our work, potentially by considering phishing websites targeting other "underrepresented" geographical areas in the PWD context**. We release all of our resources [2].

5.9 Feature Extraction

An important part of our evaluation is the feature extraction procedure, which we implement by using the open source artifact of SpacePhish [59] as basis. Such a tool extracted 57 features, but we found that two of these (i.e., *URL_fakeHTTPS* and *URL_dataURI*) are redundant since the value is the same for all samples in our datasets, so we do not consider these for our evaluation. We then expand the remaining 55 features with ten new ones: five (i.e., *H_icpReg*, *H_icpDom*, *H_icpApp*, *H_icpCode* and *H_ecert*) are Chinese-specific that follow the guidelines in [270]; while the other five (i.e., *U_unicode*, *H_nullItem*, *H_exItem*, *U_SER*, *U_tldNum*) are new and based on best practices of prior work [130, 139, 218]. Overall, our feature extractor generates the 65 features in Table 5.3.

5.9.1 Original features

We provide a high-level description of the features in Table 5.3 that differ from those in SpacePhish [59] *due to our original insights*. We stress that these features have the same logic as [59] (i.e., the value of each feature ideally denotes whether the corresponding sample is "more likely" benign or phishing).

- *H_icpReg*. If the domain is in the Ministry of Industry and Information Technology of the Chinese government, then *H_icpReg*=0; and 1 otherwise.
- *H_icpCode*. If the website includes an ICP code and it exists in its ICP recorder (obtainable by checking the domain), then *H_icpCode*=0; and 1 otherwise.
- *H_ecert*. We capture all links in the website. If none of such links point to Trustworthy website certification platforms, then *H_ecert*=1; and 0 otherwise.
- *H_icpApp*. If the domain applicant on the ICP record of Ministry of Industry and Information Technology of the Chinese government is "enterprise", then we set *H_icpApp*=0; and to 1 otherwise.
- *H_icpDom*. If the domain is consistent with the ICP record, then *H_icpDom*=0, and 1 otherwise.
- *U_unicode*. According to [270], phishing website is more likely to use UNICODE in its URL. If true, then *U_unicode*=1, and 0 otherwise.
- *H_nullItem*. This feature extends the "HTML_nullLnkWeb" of [59]. Specifically, we leverage the guidelines by Hannousse et al. [130] and factor in also other "null" elements, such as Login forms with external actions, that are typical indicators of suspiciousness (besides just blank links of "HTML_nullLnkWeb").

- *H_exItem*. We compute this feature by counting the elements in the HTML that point to external websites. The value is an integer, which will be used to compute the feature *H_obj* (which is the equivalent of the "HTML_objectRatio" of [59]).
- *U_SER*. We follow the guidelines of [139, 218], suggesting that it is possible to use search engine results to detect phishing websites. If the URL matches any of the top-10 websites in a Google search results (by querying Google with the sample's URL), *U_SER*=0; and a 1 otherwise (likely phishing).
- *U_tldNum*. We follow the guidelines of [130], suggesting that phishing websites may have more than one top-level domain (TLD) located in another position within the URL (e.g., the subdomain). This numerical feature represents the number of TLD in the URL of the sample (this information is not captured by the "URL_TLDinPath" or "URL_TLDinSub" features included in SpacePhish [59]).

To account for the nature of Chinese websites, the following two features have been *changed* w.r.t. those in [59].

- *H_titBr*. This feature checks if the website's title includes the domain of its URL. We follow the workflow described in §5.2.1. We extract the title from the corresponding HTML tag, and then we check if it includes any Chinese words via regex and, if so, we convert the Chinese words to their pronunciation (i.e.,*pinyin*). Finally, if the domain of the URL is included in the title or in the *pinyin*, *H_titBr*=0 (likely benign); and 1 otherwise (likely phishing).
- *H_DominCopr*. We search for Chinese words in the website's copyright information, convert them to their pronunciation, and finally, we check if the website's copyright information includes the website's domain: *H_DominCopr*=0 if so, and 1 otherwise.

We publicly release our feature extractor in our repo [2].

5.9.2 Validation

Let us justify why our implemented features are appropriate and also robust for our analysis.

Robustness. We recall that many of our features rely on the ICP record: one may think that phishers may try to "spoof" such ICP record in order to trick an ML-PWD. We argue this is not simple: the ICP codes are released and verified by the issuer, and they frequently change. To give an example, consider the link provided in Fig. 5.2a (https://global.jd.com/). At the bottom of the page, there is a "business licence" as a code *which is linked to an image that shows the approved licence* (in this case, it can be viewed here and here). These licences change everytime they are renewed. (Note that we do not claim our ML-PWD to be robust against "adaptive" attackers, which is outside our scope). Moreover, we have reached out to the **maintainers of a Chinese ICP records' search tool**[21], who confirmed (i) the validity of our feature extraction approach, and (ii) that the records change frequently, making it hard for phishers to keep up.

Diversity. We designed many additional features. For an appropriate analysis, it is important to determine if the values of these features among the samples in our

[21] https://www.tianapi.com/

datasets is sparse enough to prevent the occurrence of "evaluation artifacts" that bias the results [70]. To this purpose, we extract the feature representation of all samples in our `ChiPhish` dataset (shown in Table 5.8) and we perform a quantitative analysis. We find that the resulting distribution makes it "challenging" for an ML model to classify a sample on the basis of a single feature. For instance, although all benign webpages have *H_icpDom*=0, the same holds for 97% of phishing samples. Furthermore, while 98% of phishing samples have *H_ecert*=1, the same holds for 83% of benign samples. These results suggest that our testbed represents a reliable way to test the proficiency of state-of-the-art ML-PWD proposed in research.

TABLE 5.8: Distribution of our Chinese-specific features values (0s and 1s) among the samples (benign and phish) of `ChiPhish`.

Feature	Benign		Phishing	
	# 0s	# 1s	# 0s	# 1s
H_icpApp	1024	31	110	455
H_icpDom	1055	0	547	18
H_icpReg	1054	1	474	91
H_icpCode	533	522	518	47
H_ecert	183	872	7	558

Finally, we report in Table 5.9 a summary of the websites included in our `ChiPhish` dataset.

TABLE 5.9: **Summary of websites included in `ChiPhish`**. We only provide examples of *benign* websites (to protect readers).

Category	#Benign	#Phishing	Example
eCommerce	174	150	1688.com
finance	59	63	boc.cn
education	121	27	eol.cn
government	2	15	cwl.gov.cn
health	23	26	99.com.cn
email	10	13	163.com
information	267	79	labs.zol.com.cn
news	96	6	thepaper.cn
search engine	22	4	360.cn
social	35	12	weixin.qq.com
entertainment	247	83	kuwo.cn
other	0	87	n/a

5.10 Complete Evaluation Results

We report the complete results of our assessment (§5.5).

5.10.1 State-of-the-art ML-PWD

First, we report in Table 5.10 the *runtime* (in seconds) for training and testing our ML-PWD (on F_c).

TABLE 5.10: Runtime (seconds) to train and train our ML-PWD (using F_c) on our datasets. Cells report the avg (std) across 10 trials.

Alg.	WstPhish		EngPhish		ChiPhish	
	train (80%)	*test* (20%)	*train* (80%)	*test* (20%)	*train* (80%)	*test* (20%)
RF	1.2±0.093	0.06±0.003	1.37±0.146	0.06±0.004	0.19±0.026	0.01±0.003
LR	0.39±0.151	0.0006±0.0	0.768±0.2672	0.00084±0.00035	0.122±0.0619	0.0021±0.0014
DT	0.07±0.005	0.0006±0.0001	0.0656±0.01156	0.00069±0.00015	0.008±0.0011	0.0013±0.0001
KNN	0.0012±0.00013	0.2796±0.0055	0.0015±0.00019	0.47866±0.01578	0.001±0.0001	0.0174±0.0032
MLP	7.17±1.433	0.0018±0.0007	16.0134±11.42109	0.00287±0.00167	5.9±1.9532	0.0026±0.0015
CNN	695.23±3.74	0.36±0.0171	241.87±15.13	0.47±0.042	28.78±3.242	0.23±0.13
GB	4.58±0.28	0.01±0.0005	5.96±0.57	0.011±0.001	0.279±0.0384	0.002±0.0002
AB	1.73±0.13	0.08±0.0043	3.44±0.41	0.16±0.01	0.27±0.034	0.018±0.0023
SVM	2.57±0.196	0.056±0.0028	2.45±0.44	0.014±0.001	0.07±0.024	0.002±0.0002

Then, we report the detailed *tpr* and *tnr* of all the state-of-the-art ML-PWD in Tables 5.11, 5.12, 5.13 (for F_c, F_h, F_u, respectfully). Specifically, the rows of these tables denote a specific ML-PWD, identified by its learning algorithm (Alg.) and training dataset (Train 80%). The values report the average performance (and std. dev, averaged over 10 trials) of the corresponding ML-PWD on the test partition of each 'language' dataset. We report at the bottom-right the average (and std) across all algorithms.

TABLE 5.11: Performance of our custom-developed ML-PWD analyzing the F_c feature set (URL+HTML). RF is the best *Alg*.

Alg.	Train 80%	WstPhish 20%		EngPhish 20%		ChiPhish 20%		Alg.	Train 80%	WstPhish 20%		EngPhish 20%		ChiPhish 20%	
		tpr	*tnr*	*tpr*	*tnr*	*tpr*	*tnr*			*tpr*	*tnr*	*tpr*	*tnr*	*tpr*	*tnr*
RF	WstPhish	0.99±0.003	0.97±0.006	0.9±0.011	1.0±0.001	0.37±0.038	0.97±0.006	*CNN*	WstPhish	0.98±0.003	0.97±0.008	0.88±0.014	0.99±0.003	0.45±0.05	0.84±0.064
	EngPhish	0.99±0.002	0.7±0.014	0.98±0.001	1.0±0.001	0.71±0.02	0.55±0.025		EngPhish	0.99±0.005	0.68±0.053	0.98±0.005	1.0±0.002	0.61±0.174	0.49±0.05
	ChiPhish	0.99±0.005	0.13±0.036	0.97±0.016	0.49±0.152	0.96±0.015	0.99±0.005		ChiPhish	0.97±0.011	0.27±0.054	0.9±0.022	0.65±0.113	0.95±0.021	0.99±0.005
LR	WstPhish	0.96±0.006	0.95±0.007	0.88±0.013	1.0±0.001	0.52±0.056	0.95±0.017	*GB*	WstPhish	0.99±0.003	0.98±0.006	0.9±0.014	0.99±0.004	0.59±0.034	0.94±0.017
	EngPhish	0.99±0.002	0.65±0.021	0.98±0.004	1.0±0.001	0.73±0.04	0.52±0.019		EngPhish	0.99±0.002	0.37±0.032	0.99±0.003	1.0±0.001	0.85±0.03	0.52±0.029
	ChiPhish	0.99±0.003	0.13±0.014	0.93±0.007	0.39±0.049	0.95±0.014	0.99±0.005		ChiPhish	0.99±0.006	0.15±0.064	0.96±0.025	0.57±0.155	0.96±0.014	0.98±0.009
DT	WstPhish	0.98±0.006	0.96±0.006	0.86±0.016	0.9±0.222	0.56±0.077	0.82±0.062	*AB*	WstPhish	0.98±0.004	0.96±0.007	0.89±0.011	0.99±0.002	0.62±0.058	0.84±0.03
	EngPhish	0.98±0.007	0.41±0.075	0.99±0.003	0.99±0.002	0.76±0.067	0.42±0.046		EngPhish	0.99±0.003	0.51±0.024	0.99±0.002	1.0±0.001	0.72±0.059	0.51±0.049
	ChiPhish	0.98±0.01	0.16±0.076	0.95±0.035	0.69±0.165	0.96±0.015	0.97±0.015		ChiPhish	0.99±0.007	0.13±0.053	0.93±0.024	0.41±0.127	0.96±0.018	0.99±0.005
KNN	WstPhish	0.95±0.005	0.94±0.008	0.79±0.017	0.97±0.005	0.31±0.045	0.98±0.011	*SVM*	WstPhish	0.96±0.006	0.95±0.008	0.88±0.011	1.0±0.0	0.55±0.071	0.92±0.022
	EngPhish	0.84±0.009	0.9±0.008	0.92±0.011	0.99±0.002	0.55±0.055	0.94±0.018		EngPhish	0.99±0.003	0.59±0.031	0.99±0.004	1.0±0.001	0.81±0.04	0.49±0.018
	ChiPhish	0.99±0.004	0.09±0.011	0.97±0.008	0.2±0.029	0.9±0.022	0.97±0.015		ChiPhish	0.97±0.014	0.21±0.044	0.89±0.02	0.6±0.126	0.94±0.019	0.98±0.007
MLP	WstPhish	0.98±0.005	0.95±0.01	0.87±0.013	0.98±0.005	0.45±0.057	0.6±0.182	Avg (std)	WstPhish	0.97±0.014	0.96±0.011	0.87±0.033	0.98±0.028	0.49±0.099	0.87±0.113
	EngPhish	0.99±0.004	0.67±0.037	0.99±0.003	1.0±0.001	0.78±0.069	0.51±0.023		EngPhish	0.97±0.045	0.61±0.152	0.98±0.022	1.0±0.002	0.72±0.09	0.55±0.142
	ChiPhish	1.0±0.003	0.08±0.017	0.94±0.009	0.19±0.066	0.96±0.016	0.99±0.009		ChiPhish	0.98±0.007	0.15±0.055	0.94±0.025	0.47±0.172	0.95±0.019	0.98±0.007

5.10.2 MLSEC's ML-PWD

We report in Table 5.14 the exact *tpr* and *tnr* of each competition-grade ML-PWD of MLSEC.

TABLE 5.12: Performance of our custom-developed ML-PWD analyzing the F_h feature set (HTML only). RF is consistently the best *Alg*.

Alg.	Train 80%	WstPhish 20%		EngPhish 20%		ChiPhish 20%		Alg.	Train 80%	WstPhish 20%		EngPhish 20%		ChiPhish 20%	
		tpr	*tnr*	*tpr*	*tnr*	*tpr*	*tnr*			*tpr*	*tnr*	*tpr*	*tnr*	*tpr*	*tnr*
RF	WstPhish	0.95±0.006	0.93±0.009	0.75±0.029	0.97±0.002	0.17±0.038	0.96±0.011	*CNN*	WstPhish	0.93±0.004	0.89±0.016	0.69±0.021	0.93±0.006	0.2±0.053	0.85±0.137
	EngPhish	0.7±0.018	0.84±0.014	0.93±0.006	0.99±0.003	0.45±0.039	0.81±0.016		EngPhish	0.69±0.018	0.83±0.027	0.89±0.014	0.98±0.005	0.38±0.244	0.86±0.111
	ChiPhish	1.0±0.0	0.0±0.001	1.0±0.0	0.0±0.0	0.84±0.028	0.97±0.007		ChiPhish	1.0±0.001	0.01±0.008	1.0±0.001	0.01±0.011	0.82±0.034	0.97±0.009
LR	WstPhish	0.86±0.011	0.67±0.022	0.83±0.012	0.82±0.007	0.44±0.052	0.86±0.024	*GB*	WstPhish	0.93±0.008	0.88±0.016	0.78±0.018	0.93±0.007	0.18±0.046	0.96±0.013
	EngPhish	0.78±0.008	0.77±0.01	0.82±0.009	0.95±0.005	0.57±0.031	0.76±0.019		EngPhish	0.76±0.01	0.8±0.012	0.9±0.007	0.98±0.003	0.46±0.05	0.81±0.013
	ChiPhish	1.0±0.001	0.0±0.002	1.0±0.0	0.0±0.001	0.83±0.03	0.96±0.012		ChiPhish	1.0±0.003	0.01±0.01	1.0±0.001	0.03±0.024	0.84±0.035	0.97±0.008
DT	WstPhish	0.95±0.006	0.85±0.011	0.7±0.031	0.88±0.011	0.27±0.047	0.85±0.039	*AB*	WstPhish	0.91±0.01	0.78±0.015	0.79±0.014	0.87±0.009	0.15±0.024	0.97±0.019
	EngPhish	0.66±0.024	0.78±0.015	0.93±0.005	0.95±0.004	0.48±0.034	0.75±0.021		EngPhish	0.77±0.035	0.75±0.011	0.87±0.009	0.96±0.004	0.51±0.039	0.75±0.019
	ChiPhish	0.97±0.016	0.07±0.042	0.99±0.011	0.12±0.091	0.84±0.036	0.93±0.02		ChiPhish	1.0±0.001	0.02±0.011	1.0±0.002	0.05±0.039	0.83±0.035	0.97±0.011
KNN	WstPhish	0.92±0.01	0.84±0.015	0.72±0.014	0.88±0.01	0.13±0.019	0.97±0.015	*SVM*	WstPhish	0.84±0.01	0.77±0.02	0.79±0.014	0.87±0.009	0.15±0.024	0.97±0.019
	EngPhish	0.72±0.012	0.82±0.014	0.87±0.01	0.96±0.006	0.48±0.048	0.79±0.019		EngPhish	0.77±0.035	0.75±0.011	0.82±0.009	0.96±0.005	0.51±0.039	0.75±0.019
	ChiPhish	1.0±0.001	0.01±0.003	1.0±0.006	0.01±0.003	0.81±0.042	0.95±0.011		ChiPhish	1.0±0.001	0.02±0.011	1.0±0.002	0.05±0.039	0.81±0.033	0.97±0.007
MLP	WstPhish	0.91±0.013	0.88±0.023	0.73±0.042	0.91±0.014	0.2±0.034	0.86±0.064	Avg (std)	WstPhish	0.91±0.033	0.83±0.075	0.76±0.046	0.9±0.043	0.21±0.089	0.92±0.054
	EngPhish	0.7±0.025	0.83±0.015	0.86±0.013	0.98±0.005	0.46±0.036	0.81±0.015		EngPhish	0.73±0.042	0.8±0.032	0.88±0.037	0.97±0.012	0.48±0.049	0.79±0.034
	ChiPhish	1.0±0.001	0.0±0.003	1.0±0.0	0.0±0.002	0.82±0.032	0.96±0.006		ChiPhish	1.0±0.008	0.02±0.02	1.0±0.003	0.03±0.037	0.83±0.01	0.96±0.014

TABLE 5.13: Performance of our custom-developed ML-PWD analyzing the F_u feature set (URL only). RF is consistently the best *Alg*.

Alg.	Train 80%	*WstPhish* 20%		*EngPhish* 20%		*ChiPhish* 20%		Alg.	Train 80%	*WstPhish* 20%		*EngPhish* 20%		*ChiPhish* 20%	
		tpr	*tnr*	*tpr*	*tnr*	*tpr*	*tnr*			*tpr*	*tnr*	*tpr*	*tnr*	*tpr*	*tnr*
RF	WstPhish	0.99±0.003	0.96±0.006	0.89±0.011	0.99±0.002	0.53±0.039	0.95±0.012	*CNN*	WstPhish	0.98±0.005	0.96±0.006	0.89±0.014	0.98±0.014	0.68±0.06	0.71±0.077
	EngPhish	0.99±0.003	0.6±0.036	0.98±0.003	1.0±0.001	0.73±0.038	0.55±0.022		EngPhish	0.99±0.007	0.68±0.031	0.97±0.006	1.0±0.001	0.7±0.128	0.45±0.021
	ChiPhish	0.95±0.006	0.38±0.034	0.87±0.014	0.9±0.047	0.96±0.019	0.98±0.009		ChiPhish	0.92±0.011	0.39±0.041	0.84±0.013	0.86±0.037	0.94±0.013	0.98±0.008
LR	WstPhish	0.96±0.005	0.95±0.004	0.88±0.012	1.0±0.001	0.72±0.032	0.81±0.027	*GB*	WstPhish	0.98±0.002	0.97±0.005	0.89±0.012	0.97±0.019	0.68±0.054	0.84±0.026
	EngPhish	0.99±0.001	0.65±0.024	0.98±0.004	1.0±0.0	0.7±0.041	0.51±0.018		EngPhish	0.99±0.002	0.26±0.025	0.99±0.004	1.0±0.001	0.84±0.022	0.43±0.026
	ChiPhish	0.96±0.005	0.29±0.021	0.88±0.013	0.9±0.047	0.93±0.024	0.98±0.008		ChiPhish	0.95±0.007	0.36±0.025	0.88±0.012	0.91±0.033	0.95±0.013	0.98±0.007
DT	WstPhish	0.98±0.004	0.95±0.008	0.88±0.015	0.74±0.348	0.67±0.067	0.77±0.054	*AB*	WstPhish	0.97±0.005	0.95±0.005	0.88±0.012	0.99±0.002	0.75±0.039	0.72±0.033
	EngPhish	0.98±0.006	0.35±0.065	0.99±0.004	0.99±0.001	0.76±0.079	0.41±0.038		EngPhish	0.99±0.003	0.29±0.044	0.99±0.003	1.0±0.001	0.79±0.045	0.38±0.046
	ChiPhish	0.89±0.044	0.33±0.057	0.84±0.064	0.74±0.133	0.94±0.025	0.97±0.011		ChiPhish	0.94±0.007	0.39±0.032	0.86±0.015	0.89±0.111	0.95±0.016	0.97±0.015
KNN	WstPhish	0.95±0.005	0.93±0.008	0.83±0.019	0.97±0.006	0.6±0.032	0.83±0.034	*SVM*	WstPhish	0.96±0.005	0.95±0.005	0.88±0.012	1.0±0.001	0.71±0.03	0.86±0.025
	EngPhish	0.95±0.006	0.81±0.019	0.95±0.006	0.99±0.002	0.69±0.049	0.89±0.018		EngPhish	0.99±0.002	0.48±0.048	0.98±0.004	1.0±0.001	0.77±0.05	0.44±0.035
	ChiPhish	0.93±0.007	0.44±0.057	0.87±0.016	0.84±0.027	0.91±0.017	0.97±0.013		ChiPhish	0.92±0.013	0.54±0.042	0.85±0.018	0.92±0.029	0.91±0.02	0.96±0.015
MLP	WstPhish	0.98±0.004	0.96±0.009	0.88±0.016	1.0±0.002	0.55±0.039	0.84±0.04	Avg (std)	WstPhish	0.97±0.013	0.95±0.011	0.88±0.019	0.96±0.079	0.65±0.072	0.81±0.069
	EngPhish	1.0±0.002	0.63±0.037	0.98±0.005	1.0±0.001	0.74±0.053	0.48±0.027		EngPhish	0.99±0.012	0.53±0.181	0.98±0.011	1.0±0.002	0.75±0.047	0.51±0.144
	ChiPhish	0.95±0.013	0.27±0.025	0.88±0.014	0.8±0.037	0.94±0.018	0.98±0.01		ChiPhish	0.94±0.019	0.38±0.076	0.86±0.017	0.86±0.056	0.94±0.015	0.97±0.009

TABLE 5.14: Performance of each individual ML model (M) of the competition-grade ML-PWD considered in MLSEC.

M	WstPhish		EngPhish		ChiPhish	
	tpr	*tnr*	*tpr*	*tnr*	*tpr*	*tnr*
$m0$	0.50	0.99	0.58	0.99	0.14	0.99
$m1$	0.69	0.72	0.76	0.66	0.35	0.62
$m2$	0.47	0.99	0.52	0.99	0.14	0.99
$m3$	0.63	0.77	0.66	0.73	0.38	0.63
$m4$	0.54	0.98	0.62	0.99	0.14	0.99
$m5$	0.77	0.54	0.81	0.35	0.49	0.53
$m6$	0.51	0.98	0.54	0.99	0.15	0.99
$m7$	0.70	0.65	0.70	0.52	0.43	0.57
Avg (std)	0.60±0.104	0.83±0.169	0.65±0.097	0.78±0.236	0.28±0.140	0.79±0.203

5.11 Feature Importance for F_u and F_h

Let us extend our analysis in §5.6.1 by studying the feature ranking for the best classifiers using F_u and F_h in our three datasets: `ChiPhish`, `WstPhish`, `EngPhish`.

Analysis of F_u. We report in Fig 5.12 the top-10 features of the ML-PWD using RF (i.e., the best classifiers also for F_u, see Table 5.13). From Fig. 5.12, we see that there are five common features between `WstPhish` and `ChiPhish`, and six common features between `EngPhish` and `ChiPhish`. This is consistent with the results in Table 5.13, i.e., the classifier trained on `ChiPhish` has a higher performance when tested on `EngPhish` than on `WstPhish`. Interestingly, 'U_pageRank' is the most important feature learned by the RF trained on `ChiPhish`, being *three times* more important than the second ranked feature (i.e., 'U_ssl'). In contrast, for `WstPhish`, 'U_pageRank' is also the first-ranked feature, but it is not as dominating as on `ChiPhish`, since it has a similar importance than three other features (i.e., 'U_shortestWrdPath, U_lngWrdPath, U_avgWrdPath'): this can explain why the RF trained on `ChiPhish` has high fpr when tested on `WstPhish`. Finally, the rankings between the RF trained on `WstPhish` and `EngPhish` are strikingly similar, suggesting why they also exhibit a good performance when tested on different datasets in phonological languages (refer to Table 5.13).

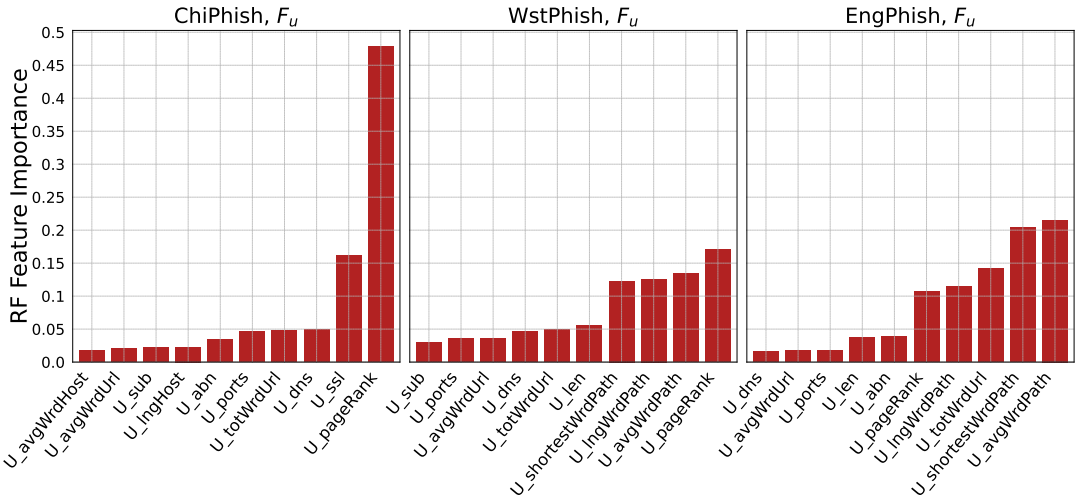


FIGURE 5.12: Feature rankings (top10) of RF (the best) using F_u.

Analysis of F_h. We report in Fig 5.13 the top-10 features of the ML-PWD using RF (which, as shown in Table 5.12, are the best classifiers also for F_h). By focusing on the ranking for `ChiPhish`, we observe that 'H_icpApp' and 'H_icpCode' are the most important features—both of which are Chinese-specific features. Surprisingly, the first feature (i.e., 'H_icpApp') is *ten times* more important than the second (i.e., 'H_icpCode'): this can explain why – despite all three classifiers sharing some features in the respective top10 (six are common between `ChiPhish` and `EngPhish`, whereas five for `ChiPhish` and `WstPhish`) – they exhibit different performance when tested on websites of a different language group. Finally (and similarly to the RF analysing F_u), there are nine common features among the RF trained on `WstPhish` and those trained on `EngPhish`: this can explain why these classifiers perform similarly even on samples of a different dataset.

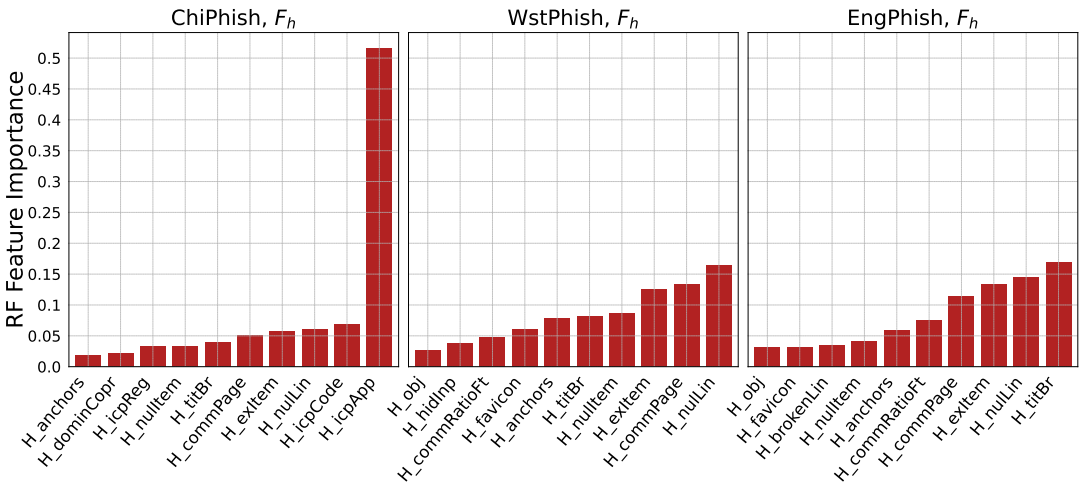


FIGURE 5.13: Feature rankings (top10) of RF (the best) using F_h.

5.12 The case of image-based PWD

We provide more details on image-based PWD. Our goal is presenting theoretical and empirical evidence that these detection approaches are *inappropriate* to explore the gap between Chinese and Western PWD. We begin by providing an overview of image-based PWD (App. 5.12.1). Next, we elucidate the shortcomings of well-known image-based PWD (App. 5.12.2), and then practically demonstrate their limitations (App. 5.12.3). Finally, we perform an original experiment showcasing a negative result (§5.12.4).

5.12.1 How do image-based PWD work?

Background. According to Corona et al. [97], PWD approaches can be divided in two categories: target *dependent*, and target *independent*. The former aim to detect phishing samples that focus on a specific target, whereas the latter seek to detect phishing without making any assumption whatsoever. For instance, the ML-PWD considered in our evaluation are all target independent: after training on a broad set of benign and phishing samples, they aim to infer whether any 'test' sample is benign or phishing. In contrast, state-of-the-art image-based PWD mostly follow a target dependent[22] approach (even in practice, e.g., [55, 109]). We are not aware of target independent PWD that use images: interestingly, Marcha et al. [184] propose ML-PWD that relies on various features (most of which overlap with ours in Table 5.3), and despite stating that screenshots are an "information source", the proposed features *do not use the screenshot*.

Target dependent PWD. These approaches focus on catching phishing websites that "target a specific brand". The intuition is that most phishing attacks try to lure their victims on (malicious) websites that resemble a reputable brand. In particular, instead of trying to infer whether a website is benign or phishing, these approaches seek to identify whether a website (or a part of it) is "similar" to another website (or a part of it) that is known to be benign. If this is true, then this finding is used to verify whether other elements of the website (e.g., its domain) match with those of the known brand.

[22] Some target-dependent PWD do not use images (e.g., [241]).

Example. The authors of PhishPedia [174] focus on the *logo*: after extracting the logo of a given webpage, [174] checks if the extracted logo matches one among the brands most targeted by phishing (e.g., PayPal). If there is no match, then the page is considered as legitimate (to avoid raising false positives). Otherwise, if a match is found (i.e., if the webpage has a logo similar to the real PayPal's), then [174] checks if the website's domain is the same as the one of the legitimate website (i.e., PayPal's): if the domain matches, then the page is legitimate (i.e., it is a page by PayPal); otherwise, the page is phishing (i.e., it is a fake page that is trying to mimic PayPal).

The reason of this two-step approach is due to *efficiency*. Indeed, querying third-party websites for domain is expensive, so it is only done if there is risk that the page is actively trying to mimic a well-known website [164]. Abundant works have proposed target dependent approaches reliant on visual similarity. Notable examples include the seminal work by Fu et al. [118], and the one by Geng et al. [121] focusing on *favicons*. More recently, we mention [44, 97, 180]. Unfortunately, the main limitation of these approaches is that they only work if the phishing webpage tries to resemble one of the targeted brands—which is typically referred to as "protected set" (PS).

5.12.2 Shortcomings of visual PWD: a case study

Problem. Image-based PWD are trendy in research, and are now being deployed also in practice [55, 109]. However, it is almost paradoxical that their biggest strength is also their main weakness. Indeed, to meet "operational" requirements, PWD must be fast: a user is not willing to wait seconds before their browser renders a given webpage *just because there is a risk of such a webpage being phishing*. Consequently, in a very short time-frame, a given PWD that employs (target dependent) image-based techniques must: *(i)* capture the screenshot of a website; *(ii)* extract the relevant information (e.g., the logo); *(iii)* make a pairwise comparison of such information with each element in the PS[23]; *(iv)* if a match is found,[24] check the domain; *(v)* after receiving the response, decide whether to block the webpage or not. This long set of operations is computationally expensive, and – to make such an analysis feasible – the PS typically includes around 200 brands [174]. Although we acknowledge that phishers tend to target well-known brands, **these methods will fail *by design* to detect** any phishing webpage that targets a brand not included in the PS.

Case Study. Let us link the information provided insofar to the problem tackled by our paper: the gap between Chinese and Western PWD. To provide evidence that *existing* (target dependent) PWD reliant on visual similarity are inappropriate for Chinese websites, we perform an in-depth look at the brands included in the PS of some well-known works. We do so by asking ourselves the two ancillary questions (AQ):

AQ1: how many of these brands are Chinese?

AQ2: how many of these Chinese brands are in the top30 Chinese websites?[25]

The rationale is that if these methods entail many (top-visited) Chinese websites in their PS, then these methods would be (somewhat) effective to counter Chinese

[23] Note that for each "protected website" there may be multiple elements associated to it (e.g., multiple logos are associated to PayPal).

[24] According to the co-authors of [55], the DNS query is done only after determining which brand is the one most likely associated to the given webpage, i.e., the PS must *always* be checked in its entirety.

[25] We take the top30 Chinese websites from [31] (in June 2023).

phishing websites. Unfortunately, the results of this case study, shown in Table 5.15, reveal that this is not the case.

TABLE 5.15: We scrutinize how many brands included in the datasets of visual PWD are from China. (N/A=data not public)

Work	PS size	# Chinese in PS (AQ1)	# top30 Chinese in PS (AQ3)
Fu [118]	8	1	0
Geng [121]	81	N/A	N/A
Corona [97]	1012	2	0
Dalgic [101]	14	1	0
Dooremaal [252]	8	N/A	N/A
Abdelnabi [44]	155	3	0
Lin [174]	181	5	1
Liu [180]	277	5	1
Apruzzese [55]	40	1	0

Results. We can see that most existing approaches have a PS with variable size, spanning between less than 10 to few hundreds brands (the exception is DeltaPhish [97], which focuses on "compromised websites" and has a slightly different focus). However, the corresponding PS have *at most five Chinese brands* in them, and none of these are included in the top30 Chinese websites. To provide further evidence, let us focus on VisualPhishNet [44] and PhishPedia [174] (and also PhishIntention [180]): the former has only 3 Chinese brands (Alibaba, Aliexpress, made-in-china), whereas the latter has 5 (Alibaba, SFexpress, Netease, made-in-china, global sources HK). This means that, at best, the corresponding PWD models can detect only Chinese phishing websites mimicing those of these six brands. However, we make two interesting observations (which we explain through Figs. 5.14):

- *Five out of these six brands are not the top30 Chinese websites.*[26] This is because Chinese websites that are also visited in the West have a different domain: for instance, "alibaba.com" (included in [174]) is less popular than "1688.com" (not included in [174]) in China—despite referring to the exact same brand.
- *The visual content in these datasets has a mismatch between the Chinese and Western versions of a brand.* For instance, [174] includes the logo for "chinese.alibaba.com" (Fig. 5.14a) but not the one for the Western version of Alibaba (i.e., "alibaba.com", Fig. 5.14b) nor the one for 1688 (Fig. 5.14c).

This means that these approaches are very unlikely to work in a "cross-language" setting (even if the corresponding PS includes some Chinese brands—since they are tailored for the Western version of such websites).

TAKEAWAY: Image-based PWD only work against phishing websites that try to mimic a shortlist of well-known brands. Unfortunately, most existing methods do not include Chinese brands in such a shortlist.

[26] The exception is NetEase, which is included in [174, 180].

(A) chinese.alibaba.com (B) alibaba.com (C) 1688.com

FIGURE 5.14: Logos of three versions of the same brand (in 2023).

5.12.3 Practical Demonstration

To further demonstrate that existing target dependent PWD reliant on visual similarity are "useless" to tackle our RQ, we perform a hard experiment with the state-of-the-art work by Liu et al., PhishIntention [180].

We take the exact implementation of PhishIntention (the code is publicly available), and we use it to analyse the visual representation of the phishing websites contained in our `ChiPhish` (recall that we store the screenshot of all samples in our datasets; see §5.3.2). Out of the 565 phishing samples, 561 (99,3%) trigger a "no target" response by PhishIntention: they are too different from any sample included in the PS and are hence flagged as benign (i.e., they evade detection). The remaining 4 trigger some similarity: 3 are (phishing) webpages that mimic the Chinese version of Apple, whereas 1 is mimicking Netease (all of which are brands included in PhishIntention's PS). However, the similarity of these is: 0.69, 0.84, 0.84, 0.72: all such values are *below the threshold* (θ=0.87) that would induce PhishIntention to proceed with the domain checking (and which would lead to a "phishing" output). Hence, these webpages are also classified as benign.[27]

Takeaway. None of the 565 phishing samples in `ChiPhish` are detected by PhishIntention [180].

Intriguingly, [180] was published in[28] 2022, and our Chinese samples in `ChiPhish` were also collected in 2022.

5.12.4 Negative result: a target INdependent image-based PWD (original experiment)

Motivation. Insofar, we have covered image-based PWD reliant on target dependent approaches. A question arises: "what about target INdependent PWD that use visual similarity?". To the best of our knowledge, *there is no paper that managed to do so effectively*. The reason is that, even by leveraging the capabilities of deep learning, it is difficult to design a PWD that can capture the nuances of benign/phishing websites just by, e.g., looking at its screenshot—given the immense variability that modern websites tend to have. Nonetheless, to provide an additional proof that image-based ML-PWD are still immature for "target independent" PWD – and hence inappropriate to investigate our main RQ (§5.5) – we perform an original proof-of-concept experiment.

Setup. We seek to develop an image-based PWD that leverages deep learning (DL) to discriminate benign from malicious webpages—i.e., a binary classification problem. For this purpose, we rely on our three datasets (§5.3)) and, specifically, on the screenshots of each webpage included therein. We chose two well-known DL algorithms as decision component: VGG16 [232] (we add dropout layers to improve

[27] We had 20 more phishing samples that mimic the Chinese Apple, but they also yielded "no target" (i.e., they also evaded [180]).

[28] We can technically re-train [180] on a different set of brands; however, according to Liu et al. [180], it takes 24h of computing on a Tesla V100 GPU to train these models.

generalization) and a CNN; we provide the exact implementation in our repository. We partition our datasets in train:test with the usual 80:20 split, and train and test each model on the same "language". We measure the performance with the *tpr* and *tnr*. We repeat this assessment 5 times. We report the detection results in Table 5.16, and the runtime in Table 5.17.

TABLE 5.16: Performance of VGG and CNN when used as binary classifiers to analyze the screenshot of a webpage in our datasets.

M	WstPhish		EngPhish		ChiPhish	
	tpr	*tnr*	*tpr*	*tnr*	*tpr*	*tnr*
CNN	$0.60_{\pm 0.021}$	$0.40_{\pm 0.037}$	$0.28_{\pm 0.020}$	$0.74_{\pm 0.028}$	$0.33_{\pm 0.054}$	$0.68_{\pm 0.022}$
VGG	$1.00_{\pm 0.000}$	$0.00_{\pm 0.000}$	$0.00_{\pm 0.000}$	$1.00_{\pm 0.000}$	$0.00_{\pm 0.000}$	$1.00_{\pm 0.000}$

TABLE 5.17: Runtime (s) to train/test VGG and CNN on our datasets. We train each model for 20 epochs (on a Tesla V100).

M	WstPhish		EngPhish		ChiPhish	
	train	test	train	test	train	test
CNN	4213.8	55.1	23048.5	298.2	2507.4	105.5
VGG	7228.9	849.9	22583.9	286.8	2185.3	26.2

Results. From these (negative) results, we can see that these DL models are terrible at discriminating benign from malicious webpages. Indeed, the performance is always skewed, showing either a perfect *tpr* but null *tnr* (and vice-versa) for VGG16; or just an unacceptably low *tpr* or *tnr* for the CNN. Furthermore, both the train and test runtime is much higher compared than our "feature-based" models (cf. Table 5.10 with Table 5.17): for instance, on WstPhish, the CNN analyzing the screenshot requires 70m to train (on GPU), whereas the CNN analyzing F_c requires 11m (on CPU). Put simply, image-based PWD that are not target-dependent are not yet ready for practical deployment—which is why we did not include similar methods in the main evaluation (§5.5).

TAKEAWAY: Image-based PWD that perform binary classification (via the screenshot) are still immature. This can be an avenue for future research.

5.13 Comparison with SpacePhish

We find it instructive to assess the performance of the vanilla version of the ML-PWD developed in SpacePhish [59]. Recall that the ML-PWD we considered in our evaluation analyse *(i)* 55 features from [59] and *(ii)* 10 additional features suggested by reputable prior work [130, 139, 218, 270]. Hence, we question whether these 10 'extra' features provide any substantial advantage w.r.t. those employed in [59]. Given that our ML-PWD use additional features (including Chinese-specific ones by [270]), we expect some form of improvement when a ML-PWD is tested on websites from the same dataset; however, the additional information may lead to overfitting.

Setup. We take the *exact* feature extractor of SpacePhish (from their repository [39]), and we use it to generate the feature representation of every sample in

our three language datasets, i.e., `ChiPhish`, `EngPhish`, `WstPhish`. Then, we consider the *exact* same learning algorithms (i.e., RF, CNN, LR.) used in SpacePhish. Finally, we adopt the *exact* procedure described in our workflow for answering RQ1 and RQ2 (refer to §5.5.1). We report the results of this "cross-language" assessment in Table 5.18 (for F_c), Table 5.20 (for F_u), and Table 5.19 (for F_h).

TABLE 5.18: Performance of the vanilla PWD of SpacePhish [59], analysing the corresponding F_c (trained and tested on our datasets).

Alg.	Train 80%	`WstPhish` 20%		`EngPhish` 20%		`ChiPhish` 20%	
		tpr	*tnr*	*tpr*	*tnr*	*tpr*	*tnr*
RF	`WstPhish`	0.99±0.004	0.97±0.007	0.90±0.007	1.00±0.001	0.37±0.052	0.97±0.010
	`EngPhish`	0.98±0.004	0.70±0.020	0.98±0.004	1.00±0.000	0.71±0.034	0.53±0.034
	`ChiPhish`	0.94±0.007	0.43±0.016	0.86±0.011	0.93±0.02	0.95±0.019	0.99±0.006
LR	`WstPhish`	0.96±0.005	0.95±0.004	0.88±0.017	1.00±0.001	0.66±0.046	0.86±0.047
	`EngPhish`	0.99±0.002	0.68±0.021	0.98±0.003	1.00±0.001	0.72±0.042	0.51±0.029
	`ChiPhish`	0.94±0.005	0.50±0.020	0.84±0.011	0.86±0.019	0.94±0.016	0.98±0.010
CNN	`WstPhish`	1.00±0.002	0.99±0.002	0.88±0.013	0.98±0.007	0.50±0.045	0.90±0.022
	`EngPhish`	0.98±0.006	0.54±0.072	1.00±0.002	1.00±0.000	0.79±0.048	0.49±0.042
	`ChiPhish`	0.93±0.007	0.45±0.026	0.85±0.013	0.90±0.031	0.93±0.020	0.98±0.006

TABLE 5.19: Performance of the vanilla PWD of SpacePhish [59], analysing the corresponding F_h (trained and tested on our datasets).

Alg.	Train 80%	`WstPhish` 20%		`EngPhish` 20%		`ChiPhish` 20%	
		tpr	*tnr*	*tpr*	*tnr*	*tpr*	*tnr*
RF	`WstPhish`	0.94±0.006	0.90±0.008	0.69±0.023	0.93±0.004	0.32±0.058	0.87±0.019
	`EngPhish`	0.59±0.023	0.84±0.020	0.90±0.011	0.98±0.004	0.26±0.055	0.87±0.017
	`ChiPhish`	0.42±0.042	0.75±0.029	0.38±0.034	0.62±0.038	0.54±0.045	0.87±0.022
LR	`WstPhish`	0.86±0.007	0.63±0.015	0.82±0.014	0.74±0.007	0.61±0.036	0.61±0.030
	`EngPhish`	0.61±0.010	0.79±0.009	0.68±0.013	0.94±0.003	0.44±0.044	0.79±0.017
	`ChiPhish`	0.44±0.025	0.79±0.017	0.40±0.019	0.59±0.031	0.45±0.028	0.88±0.020
CNN	`WstPhish`	0.94±0.005	0.91±0.008	0.65±0.017	0.88±0.012	0.35±0.066	0.85±0.025
	`EngPhish`	0.62±0.046	0.86±0.027	0.90±0.011	0.98±0.003	0.28±0.051	0.87±0.025
	`ChiPhish`	0.50±0.071	0.70±0.051	0.44±0.001	0.61±0.043	0.48±0.042	0.87±0.027

Results. Many insightful observations can be drawn by comparing these results with those of our main evaluation (shown in Tables 5.11, 5.12, 5.13). Let us focus on the most significant ones, i.e., those entailing F_c. First, as expected, each classifier of "our" ML-PWD tends to have a slightly superior performance (w.r.t. the vanilla ones in SpacePhish)[29] when tested on samples coming from the same language dataset; the best improvement is on the ML-PWD using LR (which is the learning algorithm allegedly used by Google [171]). However, we also note an intriguing phenomenon: the classifiers in SpacePhish, when trained on `ChiPhish` have a remarkably better performance when tested on `EngPhish` (w.r.t. the "enhanced" variant we used in our main evaluation—see Table 5.11). As an example, the vanilla RF of SpacePhish has a 0.93 *tnr*, whereas ours has 0.49 (even though ours has a 0.97 *tpr* against the 0.86 of

[29] Such improvement is statistically significant, e.g., a Welch t-test entailing both the *tpr* and *tnr* achieved by RF, LR, CNN trained and tested on `ChiPhish` and analysing F_c reveals that $p < 0.001$, therefore our variants are different (i.e., better) than SpacePhish's.

TABLE 5.20: Performance of the vanilla PWD of SpacePhish [59], analysing the corresponding F_u (trained and tested on our datasets).

Alg.	Train 80%	WstPhish 20%		EngPhish 20%		ChiPhish 20%	
		tpr	*tnr*	*tpr*	*tnr*	*tpr*	*tnr*
RF	WstPhish	$0.98_{\pm 0.004}$	$0.96_{\pm 0.006}$	$0.90_{\pm 0.007}$	$0.99_{\pm 0.001}$	$0.50_{\pm 0.035}$	$0.95_{\pm 0.011}$
	EngPhish	$0.99_{\pm 0.003}$	$0.63_{\pm 0.027}$	$0.98_{\pm 0.004}$	$1.00_{\pm 0.001}$	$0.73_{\pm 0.046}$	$0.54_{\pm 0.034}$
	ChiPhish	$0.95_{\pm 0.006}$	$0.38_{\pm 0.025}$	$0.88_{\pm 0.009}$	$0.88_{\pm 0.044}$	$0.96_{\pm 0.018}$	$0.99_{\pm 0.005}$
LR	WstPhish	$0.96_{\pm 0.006}$	$0.94_{\pm 0.008}$	$0.88_{\pm 0.021}$	$1.00_{\pm 0.001}$	$0.72_{\pm 0.019}$	$0.80_{\pm 0.073}$
	EngPhish	$0.99_{\pm 0.002}$	$0.68_{\pm 0.019}$	$0.98_{\pm 0.004}$	$1.00_{\pm 0.001}$	$0.75_{\pm 0.048}$	$0.49_{\pm 0.031}$
	ChiPhish	$0.95_{\pm 0.007}$	$0.43_{\pm 0.020}$	$0.88_{\pm 0.008}$	$0.93_{\pm 0.013}$	$0.93_{\pm 0.022}$	$0.98_{\pm 0.008}$
CNN	WstPhish	$0.99_{\pm 0.002}$	$0.97_{\pm 0.007}$	$0.89_{\pm 0.012}$	$0.98_{\pm 0.014}$	$0.65_{\pm 0.073}$	$0.73_{\pm 0.046}$
	EngPhish	$0.99_{\pm 0.007}$	$0.47_{\pm 0.107}$	$0.99_{\pm 0.004}$	$1.00_{\pm 0.001}$	$0.78_{\pm 0.071}$	$0.42_{\pm 0.046}$
	ChiPhish	$0.94_{\pm 0.008}$	$0.38_{\pm 0.048}$	$0.86_{\pm 0.011}$	$0.85_{\pm 0.046}$	$0.94_{\pm 0.014}$	$0.98_{\pm 0.008}$

SpacePhish). Finally, our RF and LR classifiers using F_c tend to be better than those in SpacePhish when tested on ChiPhish.

We can hence make the following **considerations**:

- Our "improved" feature sets *(i)* employ strategies proposed by reputable prior work, and *(ii)* lead to a superior baseline performance...
- ...however, in some cases, such a higher performance comes at the expense of reduced performance (especially in terms of false positives) when analysing websites coming from a different language dataset.

In summary, this experiment confirm the "no free lunch". By sacrificing some performance, it may be possible to improve the generalizability of the PWD. Our decision to develop an *ensemble* ML models (jointly with our LaSeTo) for PWD (§5.6.3) is inspired also by this result.

Chapter 6

Conclusion and Future Work

The widespread adoption of Machine Learning in Cybersecurity has contributed to advancements in state-of-the-art techniques across various domains, including phishing website detection. Machine learning-based approaches can extract features ignored by humans to get a more accurate detector. This dissertation primarily studies the security aspects of ML-based phishing website detection and evaluates the performance of ML-based phishing detectors in a multi-language environment (i.e., China and Western).

SpacePhish (in Chapter 2) and Multi-SpacePhish (in Chapter 3) formalized 'where' attackers could insert perturbations to ML-based phishing website detectors and carried out a large evaluation of evasion attacks exploiting diverse 'spaces'. Specifically, our experiments indicate that adversarial attacks that appear in website space (i.e., WSP) are more likely to be exploited by attackers because of their cheap cost. Furthermore, our result states that an attacker introduces perturbations in multiple evasion-spaces simultaneously, which caused a sharp drop in the detection rate from 0.95 to 0. Indeed, even a 3% decrease in the detection rate of ML-PWD can be problematic when dealing with thousands of samples. Our evaluation serves as a 'benchmark' for assessing the actual harm of adversarial attacks. However, our work focuses on the evaluation of evasion attacks considered text-based (i.e., HTML and URL) phishing website detectors, and future work can extend to visual-based phishing website detectors [175, 179].

At the same time, our work in Chapter 4 described adversarial phishing webpages deceived machine learning-based phishing website detectors is also a threat to the real target-end users, since most adversarial phishing webpages have comparable effectiveness on users w.r.t. unperturbed ones. This result suggested that researchers should consider users' awareness while evaluating the influence of adversarial phishing websites, and exploring potential strategies for anti-phishing training to aid users in identifying phishing webpages should be considered in the future.

Phishing website detection is a common challenge worldwide, both in Western and China. Our work shows that the gap between Chinese and Western websites does not allow existing PWD to work well in both types. In the future, researchers should consider the effectiveness of ML-PWD in multi-language environments or design phishing website detectors specific to a certain language to improve phishing detection rates.

Bibliography

[1] The most spoken languages worldwide 2022. https://www.statista.com/statistics/266808/the-most-spoken-languages-worldwide.

[2] Our repo. https://anonymous.4open.science/r/chphish_private-FE4D.

[3] Our repository.

[4] Cyber security law of the people's republic of china, 2016. http://www.npc.gov.cn/npc/c30834/201611/270b43e8b35e4f7ea98502b6f0e26f8a.shtml.

[5] Phishing activity trends report (2016). Tech. rep., Anti-Phishing Working Group, Inc, 2016. https://docs.apwg.org//reports/apwg_trends_report_q4_2016.pdf.

[6] Cn malicious websites, 2017. https://github.com/zzhihao2017/CN-Malicious-website-list.

[7] Interet crime report. Tech. rep., Federal Bureau of Investigation, 2020. https://www.ic3.gov/Media/PDF/AnnualReport/2020_IC3Report.pdf.

[8] On artificial intelligence - a european approach to excellence and trust. Tech. rep., European Commission, 2020. https://ec.europa.eu/info/sites/info/files/commission-white-paper-artificial-intelligence-feb2020_en.pdf.

[9] China mobile security status report for the first quarter of 2021. Tech. rep., 360 secure brain, Inc, 2021. https://www.freebuf.com/articles/paper/273527.html.

[10] Interet crime report. Tech. rep., Federal Bur. of Investigation, 2021. https://www.ic3.gov/Media/PDF/AnnualReport/2021_IC3Report.pdf.

[11] International Migration Outlook 2021 – China, 2021. https://www.oecd-ilibrary.org/sites/a13d0bc2-en/index.html?itemId=/content/component/a13d0bc2-en.

[12] Phishing landscape 2021: An annual study of the scope and distribution of phishing, 9 2021. https://www.interisle.net/PhishingLandscape2021.html.

[13] S&T Artificial Intelligence and Machine Learning Strategic Plan. Tech. rep., US Department of Homeland Security, 2021. https://www.dhs.gov/sites/default/files/publications/21_0730_st_ai_ml_strategic_plan_2021.pdf.

[14] The 50th statistical report on china's internet development. Tech. rep., China Internet Network Information Center, 2022. https://www.cnnic.net.cn/NMediaFile/2022/0926/MAIN1664183425619U2MS433V3V.pdf.

[15] All Adversarial Examples Papers, Accessed in Feb. 2022. https://nicholas.carlini.com/writing/2019/all-adversarial-example-papers.html.

[16] China's Rapid Development Has Transformed Its Migration Trends, 2022. https://www.migrationpolicy.org/article/china-development-transformed-migration.

[17] Chinese malicious urls, 2022. https://github.com/JiangYanting/Chinese_Malicious_Web_Pages_Dataset_And_Detection.

[18] Cyber security law of the people's republic of china, 2022. http://www.cac.gov.cn/2022-01/04/c_1642894602182845.htm.

[19] Email cyberattacks on arab countries rise in lead to global football tournament. Tech. rep., Trellix, 2022. https://www.trellix.com/en-us/about/newsroom/stories/research/email-cyberattacks-on-arab-countries-rise.html.

[20] Interet crime report. Tech. rep., Federal Bur. of Investigation, 2022. https://www.iafci.org/app_themes/docs/Federal%20Agency/2022_IC3Report.pdf.

[21] Internet crime report. Tech. rep., FBI, 2022.

[22] Machine Learning Security Evasion Competition, Accessed in Aug. 2022. https://mlsec.io/.

[23] OpenPhish, Accessed in Dec. 2022. https://openphish.com/.

[24] Phishing activity trends report. Tech. rep., APWG, 2022. https://docs.apwg.org/reports/apwg_trends_report_q2_2022.pdf.

[25] Phishing scams surge in japan, 2022. https://mainichi.jp/english/articles/20220715/p2a/00m/0na/013000c.

[26] PhishTank, Accessed in Dec. 2022. https://phishtank.org/.

[27] Statcounter: Browser Market Share China, 2022. https://gs.statcounter.com/browser-market-share/all/china.

[28] State of the phish 2022. Tech. rep., ProofPoint, 2022. https://www.proofpoint.com/it/resources/threat-reports/state-of-phish.

[29] (TrendMicro) Massive Phishing Campaigns Target India Banks Clients, 2022. https://www.trendmicro.com/en_us/research/22/k/massive-phishing-campaigns-target-india-banks-clients.html.

[30] Building a more helpful browser with machine learning. https://blog.google/products/chrome/building-a-more-helpful-browser-with-machine-learning/, 2023.

[31] Chinaz, 2023. https://top.chinaz.com.

[32] Google Safe Browsing, 2023. https://developers.google.com/safe-browsing/.

[33] kafan, 2023. https://bbs.kafan.cn/.

[34] Languages of europe, 2023. https://en.wikipedia.org/wiki/Languages_of_Europe.

[35] Netcraft, 2023. https://www.netcraft.com/.

[36] PhishDetector, 2023. https://www.moghimi.net/phishdetector.

[37] Sensitive personal data. https://home.treasury.gov/taxonomy/term/7651, 2023.

[38] similarweb23. https://www.similarweb.com/top-websites/, Accessed in Jun. 2023.

[39] Spacephish. https://spacephish.github.io/, 2023.

[40] Usage statistics of content languages for websites, 2023. https://w3techs.com/technologies/overview/content_language.

[41] Venuseye, 2023. https://www.venuseye.com.cn/.

[42] VirusTotal, 2023. https://www.virustotal.com.

[43] Websites blocked in china, Accessed in Mar. 2023. https://en.wikipedia.org/wiki/List_of_websites_blocked_in_mainland_China.

[44] ABDELNABI, S., KROMBHOLZ, K., AND FRITZ, M. VisualPhishNet: Zero-day phishing website detection by visual similarity. In *ACM CCS* (2020).

[45] ACHARYA, B., AND VADREVU, P. {PhishPrint}: Evading phishing detection crawlers by prior profiling. In *USENIX Security Symposium* (2021).

[46] ADEBOWALE, M. A., LWIN, K. T., SANCHEZ, E., AND HOSSAIN, M. A. Intelligent web-phishing detection and protection scheme using integrated features of images, frames and text. *Exp. Syst. Appl.* (2019).

[47] AHMAD, H., AND ERDODI, L. Overview of phishing landscape and homographs in arabic domain names. *Security and Privacy* (2021).

[48] AL-QURASHI, R., ALEROUD, A., SAIFAN, A. A., ALSMADI, M., AND ALSMADI, I. Generating optimal attack paths in generative adversarial phishing. In *Proc. IEEE Int. Conf. Intell. Secur. Inf.* (2021).

[49] ALEROUD, A., AND KARABATIS, G. Bypassing detection of URL-based phishing attacks using generative adversarial deep neural networks. In *Proc. Int. Workshop Secur. Privacy Anal.* (2020).

[50] ALJOFEY, A., JIANG, Q., RASOOL, A., CHEN, H., LIU, W., QU, Q., AND WANG, Y. An effective detection approach for phishing websites using url and html features. *Scientific Reports 12*, 1 (2022), 1–19.

[51] ALNAJIM, A., AND MUNRO, M. An anti-phishing approach that uses training intervention for phishing websites detection. In *2009 Sixth International Conference on Information Technology: New Generations* (2009), IEEE, pp. 405–410.

[52] ALRWAIS, S., YUAN, K., ALOWAISHEQ, E., LIAO, X., OPREA, A., WANG, X., AND LI, Z. Catching predators at watering holes: finding and understanding strategically compromised websites. In *ACSAC* (2016).

[53] ALSHARNOUBY, M., ALACA, F., AND CHIASSON, S. Why phishing still works: User strategies for combating phishing attacks. *International Journal of Human-Computer Studies 82* (2015), 69–82.

[54] APRUZZESE, G., ANDERSON, H., DAMBRA, S., FREEMAN, D., PIERAZZI, F., AND ROUNDY, K. Position:"real attackers don't compute gradients": Bridging the gap between adversarial ml research and practice. In *IEEE Conference on Secure and Trustworthy Machine Learning* (2022), IEEE.

[55] APRUZZESE, G., ANDERSON, H., DAMBRA, S., FREEMAN, D., PIERAZZI, F., AND ROUNDY, K. Position:"Real Attackers Don't Compute Gradients": Bridging the Gap Between Adversarial ML Research and Practice. In *IEEE Conference on Secure and Trustworthy Machine Learning* (2023).

[56] APRUZZESE, G., ANDREOLINI, M., FERRETTI, L., MARCHETTI, M., AND COLAJANNI, M. Modeling realistic adversarial attacks against network intrusion detection systems. *ACM Digital Threats: Res. and Practice* (2021).

[57] APRUZZESE, G., AND COLAJANNI, M. Evading botnet detectors based on flows and random forest with adversarial samples. In *Proc. IEEE Int. Symp. Netw. Comput. Appl.* (Oct. 2018), pp. 1–8.

[58] APRUZZESE, G., COLAJANNI, M., FERRETTI, L., AND MARCHETTI, M. Addressing adversarial attacks against security systems based on machine learning. In *Proc. IEEE Int. Conf. Cyber Conflicts* (May 2019), pp. 1–18.

[59] APRUZZESE, G., CONTI, M., AND YUAN, Y. Spacephish: The evasion-space of adversarial attacks against phishing website detectors using machine learning. In *Proc. ACSAC* (2022).

[60] APRUZZESE, G., CONTI, M., AND YUAN, Y. Spacephish: The evasion-space of adversarial attacks against phishing website detectors using machine learning. In *Proceedings of the 38th Annual Computer Security Applications Conference* (2022), pp. 171–185.

[61] APRUZZESE, G., LASKOV, P., MONTES DE OCA, E., MALLOULI, W., BURDALO RAPA, L., VASILEIOS GRAMMATOPOULOS, A., AND DI FRANCO, F. The role of machine learning in cybersecurity. *ACM DTRAP* (2022).

[62] APRUZZESE, G., LASKOV, P., AND TASTEMIROVA, A. Sok: The impact of unlabelled data in cyberthreat detection. In *IEEE EuroS&P* (2022).

[63] APRUZZESE, G., PAJOLA, L., AND CONTI, M. The Cross-evaluation of Machine Learning-based Network Intrusion Detection Systems. *IEEE T. Netw. Serv. Manag.* (2022).

[64] APRUZZESE, G., AND SUBRAHMANIAN, V. Mitigating adversarial gray-box attacks against phishing detectors. *IEEE Transactions on Dependable and Secure Computing* (2022).

[65] APRUZZESE, G., AND SUBRAHMANIAN, V. Mitigating adversarial gray-box attacks against phishing detectors. *IEEE TDSC* (2022).

[66] ARACHCHILAGE, N., LOVE, S., AND MAPLE, C. Can a mobile game teach computer users to thwart phishing attacks? *International Journal for Infonomics 6* (2013).

[67] ARACHCHILAGE, N. A. G., LOVE, S., AND BEZNOSOV, K. Phishing threat avoidance behaviour: An empirical investigation. *Computers in Human Behavior 60* (2016), 185–197.

[68] ARIYADASA, S., FERNANDO, S., AND FERNANDO, S. Combining long-term recurrent convolutional and graph convolutional networks to detect phishing sites using url and html. *IEEE Access* (2022).

[69] ARMSTRONG, M. E., JONES, K. S., AND NAMIN, A. S. How perceptions of caller honesty vary during vishing attacks that include highly sensitive or seemingly innocuous requests. *Human Factors 65* (2023), 275–287.

[70] ARP, D., QUIRING, E., PENDLEBURY, F., WARNECKE, A., PIERAZZI, F., WRESSNEGGER, C., CAVALLARO, L., AND RIECK, K. Dos and don'ts of machine learning in computer security. In *USENIX Secur. Symp.* (2022).

[71] AYDIN, M., AND BAYKAL, N. Feature extraction and classification phishing websites based on url. In *Conf. Commun. Netw. Secur.* (2015), IEEE.

[72] BAC, T. N., DUY, P. T., AND PHAM, V.-H. PWDGAN: Generating Adversarial Malicious URL Examples for Deceiving Black-Box Phishing Website Detector using GANs. In *Proc. IEEE Int. Conf. Machin. Learn. Appl. Netw.* (2021).

[73] BAGDASARYAN, E., AND SHMATIKOV, V. Blind backdoors in deep learning models. In *USENIX Sec. Symp.* (2021).

[74] BAHNSEN, A. C., TORROLEDO, I., CAMACHO, L. D., AND VILLEGAS, S. DeepPhish: simulating malicious AI. In *Proc. APWG Symp. Elec. Crime Res.* (2018).

[75] BAILEY, M., DITTRICH, D., KENNEALLY, E., AND MAUGHAN, D. The menlo report. *IEEE Security & Privacy 10* (2012), 71–75.

[76] BAKI, S., AND VERMA, R. M. Sixteen years of phishing user studies: What have we learned? *IEEE Transactions on Dependable and Secure Computing 20* (2023), 1200–1212.

[77] BARRACLOUGH, P. A., FEHRINGER, G., AND WOODWARD, J. Intelligent cyber-phishing detection for online. *Elsevier Comp. Secur.* (2021).

[78] BART, Y., SHANKAR, V., SULTAN, F., AND URBAN, G. L. Are the drivers and role of online trust the same for all web sites and consumers? a large-scale exploratory empirical study. *Journal of marketing 69* (2005), 133–152.

[79] BIERINGER, L., GROSSE, K., BACKES, M., BIGGIO, B., AND KROMBHOLZ, K. Industrial practitioners' mental models of adversarial machine learning. In *Proc. of SOUPS* (2022).

[80] BIGGIO, B., AND ROLI, F. Wild patterns: Ten years after the rise of adversarial machine learning. *Elsevier Pattern Recogn. 84* (2018), 317–331.

[81] BINSAEED, K., STRINGHINI, G., AND YOUSSEF, A. E. Detecting spam in twitter microblogging services: A novel machine learning approach based on domain popularity. *Int. J. Adv. Comput. Sci. Appl* (2020).

[82] BOENISCH, F., BATTIS, V., BUCHMANN, N., AND POIKELA, M. "I Never Thought About Securing My Machine Learning Systems": A Study of Security and Privacy Awareness of Machine Learning Practitioners. In *ACM Mensch und Computer*. 2021.

[83] BORTOLAMEOTTI, R., VAN EDE, T., CASELLI, M., EVERTS, M. H., HARTEL, P., HOFSTEDE, R., JONKER, W., AND PETER, A. Decanter: Detection of anomalous outbound http traffic by passive application fingerprinting. In *ACSAC* (2017).

[84] BRAUN, B., JOHNS, M., KOESTLER, J., AND POSEGGA, J. Phishsafe: leveraging modern javascript api's for transparent and robust protection. In *ACM CODASPY* (2014).

[85] BUTNARU, A., MYLONAS, A., AND PITROPAKIS, N. Towards lightweight url-based phishing detection. *Future internet* (2021).

[86] CAPUTO, D. D., PFLEEGER, S. L., FREEMAN, J. D., AND JOHNSON, M. E. Going spear phishing: Exploring embedded training and awareness. *IEEE Security & Privacy* (2013).

[87] CARLINI, N. Poisoning the Unlabeled Dataset of {Semi-Supervised} Learning. In *USENIX Secur. Symp.* (2021).

[88] CARLINI, N., ATHALYE, A., PAPERNOT, N., BRENDEL, W., RAUBER, J., TSIPRAS, D., GOODFELLOW, I., MADRY, A., AND KURAKIN, A. On evaluating adversarial robustness. *arXiv:1902.06705* (2019).

[89] CARLINI, N., AND WAGNER, D. Defensive distillation is not robust to adversarial examples. *arXiv:1607.04311* (2016).

[90] CARLINI, N., AND WAGNER, D. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proc. of AISec* (2017).

[91] CHEN, L., YE, Y., AND BOURLAI, T. Adversarial machine learning in malware detection: Arms race between evasion attack and defense. In *Europ. Intell. Secur. Inf. Conf.* (2017).

[92] CHEN, S., FAN, L., CHEN, C., XUE, M., LIU, Y., AND XU, L. Gui-squatting attack: Automated generation of android phishing apps. *IEEE TDSC* (2019).

[93] CHENG, B., MING, J., FU, J., PENG, G., CHEN, T., ZHANG, X., AND MARION, J.-Y. Towards paving the way for large-scale windows malware analysis: Generic binary unpacking with orders-of-magnitude performance boost. In *Proc. ACM CCS* (2018).

[94] CHIEW, K. L., CHANG, E. H., TAN, C. L., ABDULLAH, J., AND YONG, K. S. C. Building standard offline anti-phishing dataset for benchmarking. *International Journal of Engineering & Technology 7*, 4.31 (2018), 7–14.

[95] CHIEW, K. L., TAN, C. L., WONG, K., YONG, K. S., AND TIONG, W. K. A new hybrid ensemble feature selection framework for machine learning-based phishing detection system. *Information Sciences* (2019).

[96] CHU, W., ZHU, B. B., XUE, F., GUAN, X., AND CAI, Z. Protect sensitive sites from phishing attacks using features extractable from inaccessible phishing urls. In *Int. Conf. Commun.* (2013), IEEE.

[97] CORONA, I., BIGGIO, B., CONTINI, M., PIRAS, L., CORDA, R., MEREU, M., MUREDDU, G., ARIU, D., AND ROLI, F. Deltaphish: Detecting phishing webpages in compromised websites. In *ESORICS* (2017).

[98] CORONA, I., BIGGIO, B., CONTINI, M., PIRAS, L., CORDA, R., MEREU, M., MUREDDU, G., ARIU, D., AND ROLI, F. Deltaphish: Detecting phishing webpages in compromised websites. In *Proc. of ESORICS* (2017).

[99] CUJOAI. Machine learning security evasion competition (MLSEC), 2022. https://mlsec.io/.

[100] DAHIRU, T. P-value, a true test of statistical significance? a cautionary note. *Annals of Ibadan postgraduate medicine 6* (2008), 21–26.

[101] DALGIC, F. C., BOZKIR, A. S., AND AYDOS, M. Phish-iris: A new approach for vision based brand prediction of phishing web pages via compact visual descriptors. In *Int. Symp. Multidiscip. Stud. Innov. Tech.* (2018).

[102] DAMBRA, S., HAN, Y., AONZO, S., KOTZIAS, P., VITALE, A., CABALLERO, J., BALZAROTTI, D., AND BILGE, L. Decoding the secrets of machine learning in malware classification: A deep dive into datasets, feature extraction, and model performance. In *ACM CCS* (2023).

[103] DARKTRACE. Machine Learning in the Age of Cyber AI. Tech. rep., 2020.

[104] DEMONTIS, A., MELIS, M., BIGGIO, B., MAIORCA, D., ARP, D., RIECK, K., CORONA, I., GIACINTO, G., AND ROLI, F. Yes, machine learning can be more secure! a case study on android malware detection. *IEEE T. Dependable Secure Comp.* (2017).

[105] DEMONTIS, A., MELIS, M., PINTOR, M., JAGIELSKI, M., BIGGIO, B., OPREA, A., NITA-ROTARU, C., AND ROLI, F. Why do adversarial attacks transfer? Explaining transferability of evasion and poisoning attacks. In *Proc. USENIX Secur. Symp.* (2019), pp. 321–338.

[106] DHAMIJA, R., TYGAR, J. D., AND HEARST, M. Why phishing works. In *Proc. of CHI* (2006).

[107] DIVAKARAN, D. M., AND OEST, A. Phishing detection leveraging machine learning and deep learning: A review. *IEEE Security & Privacy* (2022), 86–95.

[108] DOWNS, J. S., HOLBROOK, M., AND CRANOR, L. F. Behavioral response to phishing risk. In *Proc. of eCrime* (2007).

[109] DRAGANOVIC, A., DAMBRA, S., IUIT, J. A., ROUNDY, K., AND APRUZZESE, G. "do users fall for real adversarial phishing?" investigating the human response to evasive webpages. In *APWG 2023 eCrime Symposium* (2023).

[110] DUMAN, S., KALKAN-CAKMAKCI, K., EGELE, M., ROBERTSON, W., AND KIRDA, E. Emailprofiler: Spearphishing filtering with header and stylometric features of emails. In *IEEE Ann. Comp. Software Appl. Conf.* (2016).

[111] EINAV, S., LEVEY, A., PATEL, P., AND WESTWOOD, A. Epistemic vigilance online: Textual inaccuracy and children's selective trust in webpages. *British Journal of Developmental Psychology 38* (2020), 566–579.

[112] FELIX, J., AND HAUCK, C. System security: a hacker's perspective. *Interex Proceedings 1* (1987), 6–6.

[113] FENG, Z., XIANGDONG, H., JIAFU, L., ZHIHUI, G., JUN, F., AND KE, L. Method of detecting the financial phishing webpage based on svm. *J. Chongqing University of Posts and Telecommunications* (2017).

[114] FERBER, R. Order bias in a mail survey. *Journal of Marketing 17*, 2 (1952), 171–178.

[115] FISCHER-HÜBNER, S., ALCARAZ, C., FERREIRA, A., FERNANDEZ-GAGO, C., LOPEZ, J., MARKATOS, E., ISLAMI, L., AND AKIL, M. Stakeholder perspectives and requirements on cybersecurity in Europe. *Elsevier J. Inf. Secur. Appl.* (2021).

[116] FLEISS, J. L., LEVIN, B., AND PAIK, M. C. *Statistical methods for rates and proportions*. john wiley & sons, 2013.

[117] FOGG, B. J., MARSHALL, J., LARAKI, O., OSIPOVICH, A., VARMA, C., FANG, N., PAUL, J., RANGNEKAR, A., SHON, J., SWANI, P., ET AL. What makes web sites credible? a report on a large quantitative study. In *Proc. of CHI* (2001).

[118] FU, A. Y., WENYIN, L., AND DENG, X. Detecting phishing web pages with visual similarity assessment based on earth mover's distance (emd). *IEEE Transactions on Dependable and Secure Computing* (2006).

[119] GANDOTRA, E., AND GUPTA, D. An efficient approach for phishing detection using machine learning. In *Multimedia Security*. 2021.

[120] GAO, P., SHAO, F., LIU, X., XIAO, X., QIN, Z., XU, F., MITTAL, P., KULKARNI, S. R., AND SONG, D. Enabling efficient cyber threat hunting with cyber threat intelligence. In *IEEE ICDE* (2021).

[121] GENG, G.-G., LEE, X.-D., WANG, W., AND TSENG, S.-S. Favicon-a clue to phishing sites detection. In *APWG eCrime Researchers Summit* (2013).

[122] GOOGLE. Compact Language Detector v3 (CLD3), 2020.

[123] GOPAVARAM, S., DEV, J., GROBLER, M., KIM, D., DAS, S., AND CAMP, L. J. Cross-national study on phishing resilience. In *Proc. of USEC* (2021).

[124] GRESSEL, G., HEGDE, N., SREEKUMAR, A., AND DARLING, M. Feature importance guided attack: A model agnostic adversarial attack. *arXiv:2106.14815* (2021).

[125] GROSSE, K., BIERINGER, L., BESOLD, T. R., BIGGIO, B., AND KROMBHOLZ, K. " why do so?"–a practical perspective on machine learning security. *arXiv preprint arXiv:2207.05164* (2022).

[126] GROSSE, K., BIERINGER, L., BESOLD, T. R., BIGGIO, B., AND KROMBHOLZ, K. Machine learning security in industry: A quantitative survey. *IEEE Transactions on Information Forensics and Security 18* (2023), 1749–1762.

[127] GUO, Z., CHO, J.-H., CHEN, R., SENGUPTA, S., HONG, M., AND MITRA, T. SAFER: Social Capital-Based Friend Recommendation to Defend against Phishing Attacks. In *Int. AAAI Conf. Web and Social Media* (2022).

[128] GUPTA, B. B., YADAV, K., RAZZAK, I., PSANNIS, K., CASTIGLIONE, A., AND CHANG, X. A novel approach for phishing URLs detection using lexical based machine learning in a real-time environment. *Elsevier Comp. Commun.* (2021).

[129] GUPTA, P., PERDISCI, R., AND AHAMAD, M. Towards measuring the role of phone numbers in twitter-advertised spam. In *ACM AsiaCCS* (2018).

[130] HANNOUSSE, A., AND YAHIOUCHE, S. Towards benchmark datasets for machine learning based website phishing detection: An experimental study. *Elsevier Eng. Appl. Artifi. Intell.* (2021).

[131] HARUTA, S., YAMAZAKI, F., ASAHINA, H., AND SASASE, I. A novel visual similarity-based phishing detection scheme using hue information with auto updating database. In *Proc. IEEE Asia-Pacific Conf. Commun.* (2019).

[132] HASEGAWA, A. A., YAMASHITA, N., AKIYAMA, M., AND MORI, T. Why they ignore english emails: The challenges of non-native speakers in identifying phishing emails. In *SOUPS* (2021).

[133] HERZBERG, A., AND JBARA, A. Security and identification indicators for browsers against spoofing and phishing attacks. *ACM Transactions on Internet Technology 8* (2008), 1–36.

[134] HO, G., CIDON, A., GAVISH, L., SCHWEIGHAUSER, M., PAXSON, V., SAVAGE, S., VOELKER, G. M., AND WAGNER, D. Detecting and characterizing lateral phishing at scale. In *Proc. USENIX Security Symp.* (2019).

[135] HOANG, N. P., NIAKI, A. A., DALEK, J., KNOCKEL, J., LIN, P., MARCZAK, B., CRETE-NISHIHATA, M., GILL, P., AND POLYCHRONAKIS, M. How great is the great firewall? measuring china's {DNS} censorship. In *USENIX Security)* (2021).

[136] HOWELL, D. Building better data protection with siem. *Elsevier Computer Fraud & Security* (2015).

[137] HR, M. G., MV, A., ET AL. Development of anti-phishing browser based on random forest and rule of extraction framework. *Cybersec.* (2020).

[138] HU, H., JAN, S. T., WANG, Y., AND WANG, G. Assessing browser-level defense against {IDN-based} phishing. In *30th USENIX Security Symposium (USENIX Security 21)* (2021), pp. 3739–3756.

[139] HUH, J. H., AND KIM, H. Phishing detection with popular search engines: Simple and effective. *FPS 11* (2011), 194–207.

[140] IUGA, C., NURSE, J. R., AND EROLA, A. Baiting the hook: factors impacting susceptibility to phishing attacks. *Human-centric Computing and Information Sciences 6* (2016), 1–20.

[141] JAIN, A. K., AND GUPTA, B. Phish-safe: Url features-based phishing detection system using machine learning. In *Cyber Security*. 2018.

[142] JAIN, A. K., AND GUPTA, B. B. Towards detection of phishing websites on client-side using machine learning based approach. *Telecommunication Systems 68*, 4 (2018), 687–700.

[143] JAIN, A. K., AND GUPTA, B. B. Towards detection of phishing websites on client-side using machine learning based approach. *Telecom. Syst.* (2018).

[144] JAIN, A. K., AND GUPTA, B. B. A machine learning based approach for phishing detection using hyperlinks information. *J. Ambient Intell. Human. Comp.* (2019).

[145] JAKOBSSON, M. The human factor in phishing. *Privacy & Security of Consumer Information 7*, 1 (2007), 1–19.

[146] JAMPEN, D., GÜR, G., SUTTER, T., AND TELLENBACH, B. Don't click: towards an effective anti-phishing training. a comparative literature review. *Human-centric Computing and Information Sciences* (2020).

[147] JANET, B., REDDY, S., ET AL. Anti-phishing system using lstm and cnn. In *Int. Conf. Innov. Tech.* (2020).

[148] JENSEN, M. L., DINGER, M., WRIGHT, R. T., AND THATCHER, J. B. Training to mitigate phishing attacks using mindfulness techniques. *Journal of Management Information Systems 34*, 2 (2017), 597–626.

[149] JIA, J., WANG, B., CAO, X., LIU, H., AND GONG, N. Z. Almost tight l0-norm certified robustness of top-k predictions against adversarial perturbations. *Int. Conf. Learn. Repr.* (2022).

[150] JORDAN, M. I., AND MITCHELL, T. M. Machine learning: Trends, perspectives, and prospects. *Science 349*, 6245 (2015), 255–260.

[151] KETTANI, H., AND WAINWRIGHT, P. On the top threats to cyber systems. In *Proc. IEEE Int. Conf. Inf. Comp. Tech.* (Mar. 2019), pp. 175–179.

[152] KIM, D., CHO, H., KWON, Y., DOUPÉ, A., SON, S., AHN, G.-J., AND DUMITRAS, T. Security Analysis on Practices of Certificate Authorities in the HTTPS Phishing Ecosystem. In *Proc. ACM AsiaCCS* (2021).

[153] KIRDA, E., AND KRUEGEL, C. Protecting users against phishing attacks with antiphish. In *COMPSAC* (2005), IEEE.

[154] KONDRACKI, B., AZAD, B. A., STAROV, O., AND NIKIFORAKIS, N. Catching transparent phish: Analyzing and detecting mitm phishing toolkits. In *ACM Conf. Comp. Commun. Secur.* (2021).

[155] KRISHNAMURTHY, B., AND WILLS, C. E. On the leakage of personally identifiable information via online social networks. In *Proc. of WOSN* (2009).

[156] KUMAR, R. S. S., NYSTRÖM, M., LAMBERT, J., MARSHALL, A., GOERTZEL, M., COMISSONERU, A., SWANN, M., AND XIA, S. Adversarial machine learning-industry perspectives. In *IEEE Secur. Privacy Workshops* (2020).

[157] KUMARAGURU, P., RHEE, Y., SHENG, S., HASAN, S., ACQUISTI, A., CRANOR, L. F., AND HONG, J. Getting users to pay attention to anti-phishing education: evaluation of retention and transfer. In *Proc. anti-phishing working groups 2nd annual eCrime researchers summit* (2007).

[158] KUMARAGURU, P., SHENG, S., ACQUISTI, A., CRANOR, L. F., AND HONG, J. Teaching johnny not to fall for phish. *ACM Transactions on Internet Technology 10* (2010), 1–31.

[159] KUNZ, A., VOLKAMER, M., STOCKHARDT, S., PALBERG, S., LOTTERMANN, T., AND PIEGERT, E. Nophish: evaluation of a web application that teaches people being aware of phishing attacks. *Informatik 2016* (2016).

[160] LAIN, D., KOSTIAINEN, K., AND ČAPKUN, S. Phishing in organizations: Findings from a large-scale and long-term study. In *2022 IEEE Symposium on Security and Privacy (SP)* (2022), IEEE, pp. 842–859.

[161] LASTDRAGER, E., GALLARDO, I. C., HARTEL, P., AND JUNGER, M. How effective is {Anti-Phishing} training for children? In *Proc. of SOUPS* (2017).

[162] LE, H., PHAM, Q., SAHOO, D., AND HOI, S. C. Urlnet: Learning a url representation with deep learning for malicious url detection. *arXiv preprint arXiv:1802.03162* (2018).

[163] LECUN, Y., BENGIO, Y., AND HINTON, G. Deep learning. *Nature* (2015).

[164] LEE, J., XIN, Z., NG, M. P. S., SABHARWAL, K., APRUZZESE, G., AND DIVAKARAN, D. M. Attacking logo-based phishing website detectors with adversarial perturbations. In *Proc. of ESORICS* (2023).

[165] LEE, J., YE, P., LIU, R., DIVAKARAN, D. M., AND CHAN, M. C. Building robust phishing detection system: an empirical analysis. *Proc. Netw. Distrib. Syst. Symp. – MADWeb Workshop* (2020).

[166] LI, Q., GUO, Y., AND CHEN, H. Practical no-box adversarial attacks against dnns. *Advances in Neural Information Processing Systems 33* (2020), 12849–12860.

[167] LI, T., KOU, G., AND PENG, Y. Improving malicious urls detection via feature engineering: Linear and nonlinear space transformation methods. *Information Systems 91* (2020), 101494.

[168] LI, T., TANG, J., XIAO, L., AND CAI, M. Evaluation of smart library portal website based on link analysis. *Proc. Comp. Sci.* (2021).

[169] LI, X., GENG, G., YAN, Z., CHEN, Y., AND LEE, X. Phishing detection based on newly registered domains. In *Int. Conf. Big Data* (2016).

[170] LI, Y., YANG, Z., CHEN, X., YUAN, H., AND LIU, W. A stacking model using url and html features for phishing webpage detection. *Future Generation Computer Systems 94* (2019), 27–39.

[171] LIANG, B., ET AL. Cracking classifiers for evasion: a case study on the google's phishing pages filter. In *Proc. World Wide Web* (2016).

[172] LIAO, C., ZHONG, H., ZHU, S., AND SQUICCIARINI, A. Server-based manipulation attacks against machine learning models. In *ACM Conf. Data and Application Security and Privacy* (2018).

[173] LIN, E., GREENBERG, S., TROTTER, E., MA, D., AND AYCOCK, J. Does domain highlighting help people identify phishing sites? In *Proc. of CHI* (2011).

[174] LIN, Y., LIU, R., DIVAKARAN, D. M., ET AL. Phishpedia: A hybrid deep learning based approach to visually identify phishing webpages. In *Proc. USENIX Secur. Symp.* (2021).

[175] LIN, Y., LIU, R., DIVAKARAN, D. M., NG, J. Y., CHAN, Q. Z., LU, Y., SI, Y., ZHANG, F., AND DONG, J. S. Phishpedia: A hybrid deep learning based approach to visually identify phishing webpages. In *30th USENIX Security Symposium (USENIX Security 21)* (2021), pp. 3793–3810.

[176] LIU, D.-J., GENG, G.-G., JIN, X.-B., AND WANG, W. An efficient multistage phishing website detection model based on the case feature framework: Aiming at the real web environment. *Computers & Security* (2021).

[177] LIU, M., ZHANG, Y., LIU, B., LI, Z., DUAN, H., AND SUN, D. Detecting and characterizing sms spearphishing attacks. In *ACSAC* (2021).

[178] LIU, M., ZHANG, Z., ZHANG, Y., ZHANG, C., LI, Z., LI, Q., DUAN, H., AND SUN, D. Automatic generation of adversarial readable chinese texts. *IEEE TDSC* (2022).

[179] LIU, R., LIN, Y., YANG, X., NG, S. H., DIVAKARAN, D. M., AND DONG, J. S. Inferring phishing intention via webpage appearance and dynamics: A deep vision based approach. In *31st USENIX Security Symposium (USENIX Security 22)* (2022), pp. 1633–1650.

[180] LIU, R., LIN, Y., YANG, X., NG, S. H., DIVAKARAN, D. M., AND DONG, J. S. Inferring phishing intention via webpage appearance and dynamics: A deep vision based approach. In *USENIX Security* (2022).

[181] LIU, Z. Perceptions of credibility of scholarly information on the web. *Information processing & management 40* (2004), 1027–1038.

[182] LO, L., LI, W., AND YU, W. Highly-skilled migration from china and india to canada and the united states. *International Migration* (2019).

[183] MAKKAR, A., KUMAR, N., SAMA, L., MISHRA, S., AND SAMDANI, Y. An intelligent phishing detection scheme using machine learning. In *Proc. International Conference on Mathematics and Computing* (2021).

[184] MARCHAL, S., SAARI, K., SINGH, N., AND ASOKAN, N. Know your phish: Novel techniques for detecting phishing sites and their targets. In *Int. Conf. Distrib. Comput. Syst.* (2016).

[185] MINK, J., KAUR, H., SCHMÜSER, J., FAHL, S., AND ACAR, Y. "Security is not my field, I'm a stats guy": A qualitative root cause analysis of barriers to adversarial machine learning defenses in industry. In *Proc. of USENIX Security* (2023).

[186] MINK, J., LUO, L., BARBOSA, N. M., FIGUEIRA, O., WANG, Y., AND WANG, G. {DeepPhish}: Understanding user trust towards artificially generated profiles in online social networks. In *31st USENIX Security Symposium (USENIX Security 22)* (2022), pp. 1669–1686.

[187] MOHAMMAD, R. M., THABTAH, F., AND MCCLUSKEY, L. An assessment of features related to phishing websites using an automated technique. In *Int. Conf. Internet Tech. Secur. Trans.* (2012).

[188] MOHAMMAD, R. M., THABTAH, F., AND MCCLUSKEY, L. Intelligent rule-based phishing websites classification. *IET Inf. Secur.* (2014).

[189] MOHAMMAD, R. M., THABTAH, F., AND MCCLUSKEY, L. Predicting phishing websites based on self-structuring neural network. *Neur. Comp. Appl.* (2014).

[190] MONTARULI, B., DEMETRIO, L., PINTOR, M., BIGGIO, B., COMPAGNA, L., AND BALZAROTTI, D. Raze to the ground: Query-efficient adversarial html attacks on machine-learning phishing webpage detectors. In *Proc. of AISec* (2023).

[191] MOORE, T. The economics of cybersecurity: Principles and policy options. *Elsevier Int. J. Critical Infrastructure Protection* (2010).

[192] MORENO-FERNÁNDEZ, M. M., BLANCO, F., GARAIZAR, P., AND MATUTE, H. Fishing for phishers. improving internet users' sensitivity to visual deception cues to prevent electronic fraud. *Computers in Human Behavior 69* (2017), 421–436.

[193] MOWAR, P., AND JAIN, M. Fishing out the phishing websites. In *Int. Conf. Cyber Situational Awareness, Data Analytics, Assess.* (2021).

[194] MU, J., WANG, B., LI, Q., SUN, K., XU, M., AND LIU, Z. A hard label black-box adversarial attack against graph neural networks. In *ACM SIGSAC Conf. Comp. Commun. Secur.* (2021).

[195] NASR, M., BAHRAMALI, A., AND HOUMANSADR, A. Defeating {DNN-Based} traffic analysis systems in {Real-Time} with blind adversarial perturbations. In *USENIX Security Symposium* (2021).

[196] NIAKANLAHIJI, A., CHU, B.-T., AND AL-SHAER, E. PhishMon: A Machine Learning Framework for Detecting Phishing Webpages. In *Proc. IEEE Int. Conf. Intel. Secur. Inf.* (2018).

[197] NIU, Y., XIE, R., LIU, Z., AND SUN, M. Improved word representation learning with sememes. In *Ann. Meet. Ass. Comp. Linguistic* (2017).

[198] OEST, A., SAFAEI, Y., DOUPÉ, A., AHN, G.-J., WARDMAN, B., AND TYERS, K. Phishfarm: A scalable framework for measuring the effectiveness of evasion techniques against browser phishing blacklists. In *2019 IEEE Symposium on Security and Privacy (SP)* (2019), pp. 1344–1361.

[199] OEST, A., SAFAEI, Y., ZHANG, P., WARDMAN, B., TYERS, K., SHOSHITAISHVILI, Y., DOUPÉ, A., AND AHN, G.-J. Phishtime: Continuous longitudinal measurement of the effectiveness of anti-phishing blacklists. In *USENIX Security Symp.* (2020).

[200] OEST, A., ZHANG, P., WARDMAN, B., NUNES, E., BURGIS, J., ZAND, A., THOMAS, K., DOUPÉ, A., AND AHN, G.-J. Sunrise to sunset: Analyzing the end-to-end life cycle and effectiveness of phishing attacks at scale. In *Proc. USENIX Secur. Symp.* (2020).

[201] O'MARA, A., ALSMADI, I., AND ALEROUD, A. Generative adverserial analysis of phishing attacks on static and dynamic content of webpages. In *Proc. IEEE Int. Conf. Parallel Distrib. Proc. Appl.* (2021).

[202] ORUNSOLU, A., AFOLABI, O., SODIYA, S., AND AKINWALE, A. A users' awareness study and influence of socio-demography perception of anti-phishing security tips. *Acta Informatica Pragensia 7* (2018), 138–151.

[203] OZCAN, A., CATAL, C., DONMEZ, E., AND SENTURK, B. A hybrid dnn–lstm model for detecting phishing urls. *Neural Comp. Appl.* (2021).

[204] PALAN, S., AND SCHITTER, C. Prolific. ac—a subject pool for online experiments. *Journal of Behavioral and Experimental Finance 17* (2018), 22–27.

[205] PANG, R., ZHANG, X., JI, S., LUO, X., AND WANG, T. Advmind: Inferring adversary intent of black-box attacks. In *ACM Int. Conf. Knowl. Discov. Data Mining* (2020).

[206] PANUM, T. K., HAGEMAN, K., HANSEN, R. R., AND PEDERSEN, J. M. Towards adversarial phishing detection. In *Proc. USENIX Workshop Cyber Security Exp. Test* (2020).

[207] PAPERNOT, N., MCDANIEL, P., GOODFELLOW, I., JHA, S., CELIK, Z. B., AND SWAMI, A. Practical black-box attacks against machine learning. In *Proc. ACM Conf. Comput. Commun. Secur.* (2017), pp. 506–519.

[208] PAPERNOT, N., MCDANIEL, P., SINHA, A., AND WELLMAN, M. Sok: Security and privacy in machine learning. In *Proc. IEEE Europ. Symp. Secur. Privacy* (Apr. 2018), pp. 399–414.

[209] PAPERNOT, N., MCDANIEL, P., WU, X., JHA, S., AND SWAMI, A. Distillation as a defense to adversarial perturbations against deep neural networks. In *IEEE Symp. Secur. Privacy* (2016).

[210] PARSONS, K., MCCORMAC, A., PATTINSON, M., BUTAVICIUS, M., AND JERRAM, C. Phishing for the truth: A scenario-based experiment of users' behavioural response to emails. In *Proc. of SEC* (2013).

[211] PATIL, S., AND DHAGE, S. A methodical overview on phishing detection along with an organized way to construct an anti-phishing framework. In *Int. Conf. Advanced Comp. Commun. Syst.* (2019), IEEE.

[212] PENG, P., YANG, L., SONG, L., AND WANG, G. Opening the blackbox of virustotal: Analyzing online phishing scan engines. In *IMC* (2019).

[213] PHISHLABS. Financials see increase in phishing attacks, compromised sites lead staging methods in q3. https://www.phishlabs.com/blog/financials-see-\increase-in-phishing-attacks-compromised-sites-lead-staging-methods-in-q3/, 2022.

[214] PIERAZZI, F., PENDLEBURY, F., CORTELLAZZI, J., AND CAVALLARO, L. Intriguing properties of adversarial ml attacks in the problem space. In *IEEE Symp. Secur. Privacy* (2020).

[215] PRAKASH, P., KUMAR, M., KOMPELLA, R. R., AND GUPTA, M. Phishnet: predictive blacklisting to detect phishing attacks. In *InfoCOM* (2010).

[216] PURKAIT, S., KUMAR DE, S., AND SUAR, D. An empirical investigation of the factors that influence internet user's ability to correctly identify a phishing website. *Information Management & Computer Security 22* (2014), 194–234.

[217] QUIRING, E., KLEIN, D., ARP, D., JOHNS, M., AND RIECK, K. Adversarial preprocessing: Understanding and preventing {Image-Scaling} attacks in machine learning. In *USENIX Secur. Symp.* (2020).

[218] RAO, R. S., AND PAIS, A. R. Detection of phishing websites using an efficient feature-based machine learning framework. *Neural Computing and Applications 31*, 8 (2019), 3851–3873.

[219] RATHORE, H., AGARWAL, S., SAHAY, S. K., AND SEWAK, M. Malware detection using machine learning and deep learning. In *Springer Int. Conf. Big Data Analytics* (2018), pp. 402–411.

[220] ROEPKE, R., DRURY, V., PEESS, P., JOHNEN, T., MEYER, U., AND SCHROEDER, U. More than meets the eye-an anti-phishing learning game with a focus on phishing emails. In *Int. Conf. Games and Learn. Alliance* (2022).

[221] SABIR, B., BABAR, M. A., AND GAIRE, R. An evasion attack against ML-based phishing URL detectors. *arXiv:2005.08454* (2020).

[222] SAHINGOZ, O. K., BUBER, E., DEMIR, O., AND DIRI, B. Machine learning based phishing detection from urls. *Exp. Syst. Appl.* (2019).

[223] SÁNCHEZ-PANIAGUA, M., FIDALGO, E., GONZÁLEZ-CASTRO, V., AND ALEGRE, E. Impact of current phishing strategies in machine learning models for phishing detection. In *Proc. of CISIS* (2021).

[224] SARNO, D. M., MCPHERSON, R., AND NEIDER, M. B. Is the key to phishing training persistence?: Developing a novel persistent intervention. *Journal of Experimental Psychology: Applied* (2022).

[225] SCOTT, M. J., GHINEA, G., AND ARACHCHILAGE, N. A. G. Assessing the role of conceptual knowledge in an anti-phishing educational game. In *Proc. of ICALT* (2014).

[226] SHAN, S., WENGER, E., WANG, B., LI, B., ZHENG, H., AND ZHAO, B. Y. Gotta catch'em all: Using honeypots to catch adversarial attacks on neural networks. In *Proc. ACM SIGSAC Conf. Comp. Commun. Secur.* (2020), p. 67–83.

[227] SHARMA, S. R., PARTHASARATHY, R., AND HONNAVALLI, P. B. A feature selection comparative study for web phishing datasets. In *Int. Conf. Elec. Comp. Commun. Tech* (2020).

[228] SHENG, S., HOLBROOK, M., KUMARAGURU, P., CRANOR, L. F., AND DOWNS, J. Who falls for phish? a demographic analysis of phishing susceptibility and effectiveness of interventions. In *Proc. of CHI* (2010).

[229] SHIRAZI, H., BEZAWADA, B., RAY, I., AND ANDERSON, C. Adversarial sampling attacks against phishing detection. In *IFIP Ann. Conf. Data Appl. Secur. Privacy* (2019).

[230] SHUSTERMAN, A., AVRAHAM, Z., CROITORU, E., HASKAL, Y., KANG, L., LEVI, D., MELTSER, Y., MITTAL, P., OREN, Y., AND YAROM, Y. Website fingerprinting through the cache occupancy channel and its real world practicality. *IEEE TDSC* (2020).

[231] SIMKO, L., LERNER, A., IBTASAM, S., ROESNER, F., AND KOHNO, T. Computer security and privacy for refugees in the united states. In *IEEE S&P* (2018).

[232] SIMONYAN, K., AND ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[233] SINGH, P., JAIN, N., AND MAINI, A. Investigating the effect of feature selection and dimensionality reduction on phishing website classification problem. In *Int. Conf. Next-Generation Comput. Tech.* (2015), IEEE.

[234] SMEAL, W., KUMAR, Y., VISHWANATH, V., CAMP, L. J., AND ALEXEEV, A. Phishing resiliency across socio-cultural spheres: Cyrillic orthographic zone vs. the five eyes. In *ACSAC'22 Poster session* (2022).

[235] SMUTZ, C., AND STAVROU, A. Malicious PDF detection using metadata and structural features. In *Proc. ACM Ann. Comp. Secur. Appl. Conf.* (2012).

[236] SONG, F., LEI, Y., CHEN, S., FAN, L., AND LIU, Y. Advanced evasion attacks and mitigations on practical ML-based phishing website classifiers. *Int. J. Intell. Syst.* (2021).

[237] SU, J., VARGAS, D. V., AND SAKURAI, K. One pixel attack for fooling deep neural networks. *IEEE T. Evol. Comput.* (2019).

[238] SU, J., VARGAS, D. V., AND SAKURAI, K. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation 23*, 5 (2019), 828–841.

[239] SUBRAMANI, K., MELICHER, W., STAROV, O., VADREVU, P., AND PERDISCI, R. Phishinpatterns: measuring elicited user interactions at scale on phishing websites. In *IMC* (2022).

[240] TAIB, R., YU, K., BERKOVSKY, S., WIGGINS, M., AND BAYL-SMITH, P. Social engineering and organisational dependencies in phishing attacks. In *Proc. of Interact* (2019).

[241] TAN, C. L., CHIEW, K. L., WONG, K., ET AL. PhishWHO: Phishing webpage detection via identity keywords extraction and target domain name finder. *Elsevier Decis. Support Syst.* (2016).

[242] TANG, L., AND MAHMOUD, Q. H. A survey of machine learning-based solutions for phishing website detection. *Machine Learning and Knowledge Extraction* (2021).

[243] TAUSCH, A. Muslim immigration continues to divide europe: A quantitative analysis of european social survey data. *Middle East Review of International Affairs* (2016).

[244] TEMBE, R., ZIELINSKA, O., LIU, Y., HONG, K. W., MURPHY-HILL, E., MAYHORN, C., AND GE, X. Phishing in international waters: exploring cross-national differences in phishing conceptualizations between chinese, indian and american samples. In *Symp. Bootcamp Sci. Secur.* (2014).

[245] THOMAS, D. R. A general inductive approach for analyzing qualitative evaluation data. *American journal of evaluation 27*, 2 (2006), 237–246.

[246] THOMPSON, C., SHELTON, M., STARK, E., WALKER, M., SCHECHTER, E., AND FELT, A. P. The web's identity crisis: understanding the effectiveness of website identity indicators. In *Proc. of USENIX Security* (2019).

[247] TIAN, K., JAN, S. T., HU, H., YAO, D., AND WANG, G. Needle in a haystack: Tracking down elite phishing domains in the wild. In *Proc, ACM Internet Measurement Conf.* (2018).

[248] TONG, L., LI, B., HAJAJ, C., XIAO, C., ZHANG, N., AND VOROBEYCHIK, Y. Improving robustness of ml classifiers against realizable evasion attacks using conserved features. In *USENIX Secur. Symp.* (2019).

[249] TRAMÈR, F., DUPRÉ, P., RUSAK, G., PELLEGRINO, G., AND BONEH, D. Adversarial: Perceptual ad blocking meets adversarial machine learning. In *ACM Conf. Comp. Commun. Secur.* (2019).

[250] TRAMÈR, F., KURAKIN, A., PAPERNOT, N., GOODFELLOW, I., BONEH, D., AND MCDANIEL, P. Ensemble adversarial training: Attacks and defenses. In *Proc. Int. Conf. Learning Representations* (2018).

[251] TSOW, A., AND JAKOBSSON, M. Deceit and deception: A large user study of phishing. *Indiana University. 9* (2007).

[252] VAN DOOREMAAL, B., BURDA, P., ALLODI, L., AND ZANNONE, N. Combining text and visual features to improve the identification of cloned webpages for early phishing detection. In *Proc. ACM ARES* (2021).

[253] VEERAMACHANENI, K., ARNALDO, I., KORRAPATI, V., BASSIAS, C., AND LI, K. Aiˆ 2: training a big data machine to defend. In *Proc. IEEE Int. Conf. Big Data Secur. Cloud* (2016).

[254] VERMA, A. Effects of phishing on e-commerce with special reference to india. In *Int. Persp. Busin. Converg. Comput. Legal.* 2013.

[255] VERMA, R., AND DYER, K. On the character of phishing urls: Accurate and robust statistical learning classifiers. In *Proc. ACM Conf. Data Appl. Secur. Privacy* (2015).

[256] WANG, J., LI, Y., AND RAO, H. R. Overconfidence in phishing email detection. *Journal of the Association for Information Systems 17* (2016), 1.

[257] WANG, J., WANG, L., DONG, F., AND WANG, H. Re-measuring the label dynamics of online anti-malware engines from millions of samples. In *IMC* (2023).

[258] WEI, W., KE, Q., NOWAK, J., KORYTKOWSKI, M., SCHERER, R., AND WOŹNIAK, M. Accurate and fast URL phishing detector: a convolutional neural network approach. *Elsevier Comp. Netw.* (2020).

[259] WILSON, K. S., AND KIY, M. A. Some fundamental cybersecurity concepts. *IEEE Access* (2014).

[260] WRIGHT, R. T., JENSEN, M. L., THATCHER, J. B., DINGER, M., AND MARETT, K. Research note—influence techniques in phishing attacks: An examination of vulnerability and resistance. *Information systems research 25* (2014), 385–400.

[261] XIANG, G., HONG, J., ROSE, C. P., AND CRANOR, L. Cantina+: A feature-rich machine learning framework for detecting phishing web sites. *ACM T. Inf. Syst. Secur. 14*, 2 (2011), 21.

[262] XIANGDONG, H., KE, L., FENG, Z., JIAFU, L., JUN, F., AND ZHIHUI, G. Financial phishing detection method based on sensitive characteristics of webpage. *Chinese J. Netw. Inf. Secur.* (2017).

[263] XIONG, A., PROCTOR, R. W., YANG, W., AND LI, N. Is domain highlighting actually helpful in identifying phishing web pages? *Human factors 59* (2017), 640–660.

[264] XIONG, A., PROCTOR, R. W., YANG, W., AND LI, N. Embedding training within warnings improves skills of identifying phishing webpages. *Human Factors* (2019).

[265] YANG, C.-C., TSENG, S.-S., LEE, T.-J., WENG, J.-F., AND CHEN, K. Building an anti-phishing game to enhance network security literacy learning. In *Proc. of ICALT* (2012).

[266] YANG, L., ZHANG, J., WANG, X., LI, Z., LI, Z., AND HE, Y. An improved elm-based and data preprocessing integrated approach for phishing detection considering comprehensive features. *Expert Systems with Applications 165* (2021), 113863.

[267] YOON, C., KIM, K., KIM, Y., SHIN, S., AND SON, S. Doppelgängers on the dark web: A large-scale assessment on phishing hidden web services. In *The World Wide Web Conference* (2019).

[268] YOON, C., KIM, K., KIM, Y., SHIN, S., AND SON, S. Doppelgängers on the dark web: A large-scale assessment on phishing hidden web services. In *WWW* (2019).

[269] YUAN, Y., APRUZZESE, G., AND CONTI, M. Multi-spacephish: Extending the evasion-space of adversarial attacks against phishing website detectors using machine learning. *arXiv preprint: 2210.13660* (2023).

[270] ZHANG, D., YAN, Z., JIANG, H., AND KIM, T. A domain-feature enhanced classification model for the detection of chinese phishing e-business websites. *Information & Management* (2014).

[271] ZHANG, J., PAN, Y., WANG, Z., AND LIU, B. Url based gateway side phishing detection method. In *IEEE Trustcom/BigDataSE/ISPA* (2016).

[272] ZHANG, P., OEST, A., CHO, H., SUN, Z., JOHNSON, R., WARDMAN, B., SARKER, S., KAPRAVELOS, A., BAO, T., WANG, R., ET AL. Crawlphish: Large-scale analysis of client-side cloaking techniques in phishing. In *IEEE Symp. Security and Privacy* (2021).

[273] ZHANG, W., JIANG, Q., CHEN, L., AND LI, C. Two-stage elm for phishing web pages detection using hybrid features. *World Wide Web* (2017).

[274] ZHANG, X., ZENG, Y., JIN, X.-B., YAN, Z.-W., AND GENG, G.-G. Boosting the phishing detection performance by semantic analysis. In *Int. Conf. Big Data* (2017), IEEE.

[275] ZHENG, B., JIANG, P., WANG, Q., LI, Q., SHEN, C., WANG, C., GE, Y., TENG, Q., AND ZHANG, S. Black-box adversarial attacks on commercial speech platforms with minimal information. In *ACM CCS* (2021).

[276] ZURAIQ, A. A., AND ALKASASSBEH, M. Phishing detection approaches. In *Int. Conf. new Trends in Computing Sciences (ICTCS)* (2019), IEEE.

[277] ŠRNDIC, N., AND LASKOV, P. Practical evasion of a learning-based classifier: A case study. In *Proc. IEEE Symp. Secur. Privacy* (2014), pp. 197–211.