International Journal of Informatics, Information System and Computer Engineering

Phishing Website Detection Using Several Machine Learning Algorithms: A Review Paper

Alexander Veach, Munther Abualkibash*

School of Information Security and Applied Computing, Eastern Michigan University, Ypsilanti, Michigan, United States

*Corresponding Email: aveach1@emich.edu

ABSTRACTS

Phishing is one of the major web social engineering attacks. This has led to demand for a better way to predict and stop them in a commercial environment. This paper seeks to understand the research done in the field and analyse the next steps forward. This is done by focusing on what goes into the selection of proper features, from manual selection to the use of Genetic Algorithms such as ADABoost and MultiBoost. Then a look into the classifiers in use, Neural Networks and Ensemble algorithms which were prominent alongside some novel approaches. This information is then processed into a framework for cloud-based and client-based phishing website detection, alongside suggestions for possible future research and experiments that could help progress the field.

ARTICLE INFO

Article History:
Received 18 Dec 2022
Revised 20 Dec 2022
Accepted 25 Dec 2022
Available online 26 Dec 2022

Keywords:
Artificial Intelligence, Data Science, Machine Learning, Phishing.

1. INTRODUCTION

Phishing has become one of the most prevalent social engineering attacks in the digital environment. From personal accounts to corporate user accounts, all must be aware of the potential dangers of a phishing attack. This has led to an ongoing battle to prevent phishing attacks by blocking dangerous websites and communications. There are many methods to fight these attacks, with many looking to the new advancements in machine learning and artificial intelligence as a potential solution to phishing attacks. The method discussed in this paper is detecting phishing websites with machine learning algorithms.

Unfortunately, such a problem lacks a catch-all solution, which has led to

the formation of multiple different approaches to the problem. For example, one solution could suggest designing methods for on-hardware machine learning, which will limit the choice of algorithms to simpler versions but will allow for mass implementation. Other solutions could focus on offloading the classification and model to a third-party service like Microsoft Azure or Amazon Web Services, which circumvents the limitation of algorithms in exchange for another group of issues.

Including the differences in selecting features, where to gather the data, and much more there are a multitude of potential solutions with many looking for the most effective solution. The purpose of this paper is to look at the potential solutions and outline what the next steps for such research could be.

2. METHOD

To analyze the current popular solutions and implementation of anti-phishing technologies using machine learning and artificial intelligence a plethora of research was gathered from Collage Repositories and online journal sites such as JSTOR. Once the multitude of research was gathered, which amounted to 91 papers. These 91 papers were then read and analyzed, taking the classifier and methods used into account and their differences. Once that was complete, papers with relevance to the topic at hand and important for discussion were selected and used, the number of which is 14.

3. RESULTS AND DISCUSSION

3.1. The Material Used

The application of machine learning against phishing is not a new development and there has been a multitude of research done over the last few years. Especially so for phishing URLs. This is some of the relevant research that has come out in the last few years.

There is Sanchez-Paniagua et al. (2022) who focused on analysing deep learning methods compared to other methods, namely ensemble and genetic selection algorithms. In their study they found that their model of using TF-IDF + N-gram outperformed other methods by varying degrees. With the closest performers being within 0.5 points of accuracy while the weakest performers were behind as much as 10 points. The researchers also found that "...handcrafted URL features decrease their performance over time, up to 10.42% accuracy in the case of the LightGBM algorithm from the year 2016 to 2020. For this reason, machine learning methods should be trained with recent URLs to prevent substantial aging from the date of its release" (Sanchez-Paniagua et al., 2022).

Xiao et al. (2020) focused on using CNN with multi-head self-attention to determine if links were valid or phishing. By using MHSA, the researchers found better accuracy and speed compared to CNN-LTSM with a difference of 0.002 in CNN-MHSA's favour. For future work, Xiao et al. (2020) focuses on updating the model to take the HTML content into consideration to increase the accuracy further (Xiao et al., 2020).

A different direction was pursued by Suleman and Awan (2019), who focused on the use of generic algorithms such as "Yet Another Generating Genetic

Algorithm" or YAGGA. Testing it against other GAs found a 94.99% accuracy with an ID3 classifier (Suleman & Awan, 2019).

Another example of study when it comes to genetic algorithms is Subasi and Kremic (2020) who compared Adaboost and MultiBoosting when it came to testing phishing websites. The researchers found a high accuracy of 97.61% using an SVM classifier with Adaboost, however the cost of that accuracy is that SVM Adaboost reported a complexity, in seconds, of "8193.72" (Subasi & Kremic, 2020).

Another genetic algorithm study comes from Alsariera et al. (2020) who focused on using their Forest Penalizing Attributes algorithm that uses weight to deemphasize inconsequential variables. The team then compared the results to meta-learning variants of the algorithm specifically testing a bagging method and Adaboost. Of which they found that Adaboosted Forest Penalizing Attributes had an accuracy of 97%, beating the other accuracies of 96.26% for base classifier and 96.58% for bagged, and a speed where "...false alarm notifications are next to zero" (Alsariera et al., 2020).

A more unique approach is the one Chen et al. (2020) took focusing on the visual similarity of websites to determine if it is a phishing website. It does this by using wavelet hashing and Scale-Invariant Feature Transform to determine similarity. The researchers found some success when using Microsoft, Dropbox, and Bank of America as a comparison point, getting accuracy results of 98.14%, 98.61% and 99.95% respectively (Chen et al., 2020).

Another unique approach is that of Ali and Malebary (2020), who used Particle Swarm Optimization to improve detection of fraudulent phishing websites. By using the high speed PSO model the team proposes feature weighting in much the same way a genetic algorithm operates. Compared to the GA selection and weighting the team found that "...PSO-based feature weighting omitted between 7%-57% of irrelevant features" and found that classifiers using their method "...outperformed these machine learning models with applying IG, Chi-Square, Wrapper, GA-based features selection, and GA-based features weighting" (Ali & Malebary, 2020).

Another approach takes the visual analysis of websites but then combines it with a neural network classifier. This approach is what Abdelnabi et al. (2020) proposed which uses a triplicate network to compare websites to popular websites on Alexa. By using the ensemble method with neural networks, they outline a potential future path for using website matching (Abdelnabi et al., 2020).

Assefa and Katarya (2022) focused on analysing other deep learning methods and their results and compared it to Autoencoder, a form of unsupervised neural network. In the report they noted various limitations in other studies, noting issues such as non-comprehensive reports and compared their achievements to the Autoencoder method. They found that Autoencoder had an accuracy of 91.24% and that with better data mining techniques the performance could be improved (Assefa & Katarya, 2022).

Mandadi et al. (2022) focused on finding the most important features denoting three types, Domain-Based, HTML and JavaScript Based, and Address Bar Based features, with the total number of features under these three categories being considered was 17. Once

that was set, they tested the features with Random Forest and Decision Tree which gave values of 87.0% and 82.4% for accuracy respectively (Mandadi et al., 2022).

Saravanan and Subramanian (2020) used GA feature selection alongside an ARTMAP supervised neural network. ARTMAP is made up of "a pair of self organized Adaptive Resonance Theory (ART) modules ARTa and ARTb. These two modules are interconnected by an inter-ART self associative memory and an internal controller, whose objective is to maximize the predictive generalization and to minimize the predictive error. Each ART module is associated with F1 and F2 layers which act as a short term memory and a long term memory for category selection" (2020). This model also uses the Firefly Algorithm to determine which features are useful. The study found their own unique algorithm to be the best performing in all performance measures except for detection time, which SVM performed better (Saravanan & Subramanian, 2020).

Mourtaji et al. (2017) also outlines which features they believe are best suited for detection. Having five groups which are: lexical based analytics method, abnormal based feature, content-based analytics method, and an identity-based method. Alongside these features they suggest a blacklist function on-top of these features. They used a linear regression classifier and reported an accuracy of 95.5% with a false positive rate of 1.4% (Mourtaji et al., 2017).

Zhou and Zhang (2022) propose a dual-weight random forest algorithm that is "based on the combination of feature weight and decision tree weight". The proposed classifier was then tested against Random Forest, Random Forest Algorithm with Decision Tree Weight, and Dynamic Random Forest and had the highest Accuracy with a value of 94.93% which was 2.22 points higher than the next highest which was Dynamic Random Forest with 92.71 (Zhou & Zhang, 2022).

3.2. Analysis

Phishing is one of the most dangerous and effective online fraud methods in existence today. This concern has led to the search for a so-called "silver bullet" that would protect potentially affected parties from phishing attacks. Many have looked towards machine learning and artificial intelligence to create an application that, when used, would detect threats and adapt to them to create the ultimate defense. However there are many parts to consider including which classifier should be used for training, what attributes should be weighed to determine threat and which dataset is the best for training the model.

The first major question is by which metric should such a model be trained around. Should it be URL focused, should it be based upon the content of the website itself, or should it be based on the websites meta content using tools such as WHOIS. URL based analysis is simple to implement and fast to process, but lacks other information from the website which can decrease accuracy. Similarly analyzing the content of the web page alongside the URL itself takes more time to execute for the benefit of more accurate results. Some even suggest image recognition models such as Chen et al. (2020) with their visual similarity model.

Then, when it comes to weighing features, some papers suggest using

attribute selection algorithms such as ADABoost, MultiBoost, or other genetic algorithms to predict which attributes lend themselves to correct identification such as Suleman and Awan (2019), Subasi and Kremic (2020), and Alsariera et al. (2020). By using these machine proven attributes many hope to increase the efficiency of the used classification algorithms. Subasi and Kremic (2020) noted that "Adaboost achieved the superior classification accuracy, with SVM 97.61%" which beat their best accuracy single classifier result which was Random Forest which achieved "an accuracy of 97.26%". Another study done by Sanchez-Paniagua et al. (2022) reported that, when testing trained models based on data from 2016, 2017 and 2020: "...all models struggled to endure over time and their performance decreased when tested on the following years' dataset" (Sanchez-Paniagua et al., 2022). Thus showing the importance of an ever updating classification scheme.

There are many offered solutions when it comes to what classifier to use, with two of the most common answers being Neural Network classifiers and Random Forest classification. Random Forest has been found by many researchers to be their choice of classifier in the studies surveyed. Zhou et al. (2020) used a modified version of Random Forest, named Double Weighted Random Forest, and returned an accuracy 94.94% when using K-means clustering for feature selection. In studies that found other methods to be more effective such as Sanchez-Paniagua et al. the difference was only a 0.20 accuracy difference compared to LightGBM with 94.67 (Sanchez-Paniagua et al., 2022). However, some report a lower accuracy number, such as Mandadi et al. (2022) who found a reported accuracy of 82.4% with 17 features using a PhishTank dataset. This variance could be attributed to the differences in feature selection and the contents of the used datasets.

Another common solution is the use of Neural Network classifiers such as CNN, LSTM, GNN and many others. Neural Network classification is recommended similarly to Random Forest with many studies finding high accuracy when predicting malicious phishing URLs. As mentioned in the section prior, Sanchez-Paniagua et al. found that Light BGM had the highest tested accuracy of the classifiers used with static feature selection on the PIU-60K dataset (Sanchez-Paniagua et al., 2022). Other studies have noticed similar results with other neural networks, specifically those with deep learning capabilities. Xiao et al. (2020) applied multi-head self-attention, or MHSA, to a Convolution Neural Network and found an accuracy rate of 0.9834 or 98.34 percent. The study proposed more solutions to increase that number even higher with their main worry being to "decrease the input of [URL's length parameter]" (Xiao et al., 2020).

Novel application of the prior is also well-researched. With a common focus on using visual detection, to detect pages that are too close to other pages as seen in Abdelnabi et al's work (2020). In their research they proposed a model that uses three convolutional models to determine phishing or not based on the similarity to other major pages collected from Alexa. Another unique approach is Ali and Malebary (2020) who propose a model based on Particle Swarm Optimization feature weighing. Which reportedly outperformed other weighting algorithms.

Like most topics there is not a singular silver bullet, so to speak, when it comes to predicting if a website is malicious or not. Phishing methods commonly change to what is most efficient at that time which has led to a never ending conflict trying to prevent said attacks. This has led to a focus on using Genetic Algorithms or other methods to create a curated list of features. As noted by Sanchez-Paniagua et al, "compared to machine learning algorithms, both CNN models obtained better results than handcrafted features" (Sanchez-Paniagua et al., 2022). By using deep learning models, a higher level of accuracy can be maintained at the cost of more costly requirements. Neural Network classifiers by design develop a much richer identification method, upon which they layer information in a way imitating human neurons, which requires more processing power than simple classifiers such as a Decision Tree classifier. These methods, when properly trained, can generate extremely accurate results. However, this in of itself is a much more costly method requiring a higher level of processing power commonly using high-end graphics cards designed for that explicit purpose such as the Nvidia Titan V.

On the other end of the spectrum is Random Forest, or other ensemble classifiers, that instead rely on a series of classification tests to assure accuracy. Thanks to this, ensemble classifiers require less processing power and have a better success rate with less data provided. However, Random forest lacks the potential depth of learning that deep learning neural networks can possibly provide and is not adept when adapting to changes over time, as reported by Sanchez-Paniagua et al. (2022).

Then there are two further trains of thought when it comes to implementation, if the software should be designed to run off of the hardware it is installed upon or if the hardware should be run off of virtualized software through the cloud. Both have their benefits and drawbacks, as offloading the processing better works when using devices such as mobile phones and other low powered devices. However, this builds a dependency on stable connection for the service to work, and a reliance on consistent service. This then creates specifications of an infrastructure that can support such needs. While using the physical machine itself limits the potential design of the model, as it must be customized to each device or be designed to work with most devices sacrificing customization. The benefit would be reliability, as the model would only require the model that is already trained and the processing power of the device executing it. This would limit potential downtime and other server connectivity issues, but could cost more in the long run for businesses implementing this method. Another issue would be training the models in a reasonable way to adapt to changes in phishing techniques. Something which Sanchez-Paniagua et al. (2022) found as much as a 10% decrease in accuracy as malicious phishing links change.

The next most common solution was custom classifiers or unique analysis methods, or other similar methods, which made up nineteen of the ninety papers analyzed. These solutions focused on designing custom classifiers that would parse the target information, with claims that the unique solution was more effective than other common solutions. These classifiers are often similar to

ensemble methods which combine classifiers in a multilayered approach. However, some are amalgamations designed to work as a single classifier instead of the normal multileveled classification that ensemble methods use which is why they have their own category. Some of these solutions claim to have a success rate when tested of around 98 percent while others claim a much lower result. For example, Saravanan and Subramanian (2022) used a combination of a Genetic Algorithm to select important features and ARTMAP, a neural network classifier based upon the Firefly Algorithm.

There was also another group that had nineteen studies suggest its use. The deep learning methods are made up of such classifiers as CNN, DNN, GNN and their derivatives. These methods were used specifically to design evolving models that could potentially detect new attacks and adapt quickly. An issue with these studies is of course the resource intensive nature of deep learning methods. The method's resource intensive nature leaves only two options when it comes to potential implementation: require all hardware to meet the specification or offload the AI to a cloud-based solution. By requiring a dedicated GPU any company wishing to adopt will face a steep entry cost which will be a barrier to general adoption especially for major companies with tens of thousands of workers. The same is true for a Cloud based solution as any corporation that wishes to adopt such a method will undoubtedly pay fees for such usage.

Something that was noticed in many of the reports is a lack of standardization when it comes to reporting the information gained from experimentation. Several papers only reported the Accuracy without any of the other data points leaving you to extrapolate how they reached that conclusion. This issue has been noted in other papers such as "Intelligent Phishing Website Detection Using Deep Learning", where Assefa and Katarya (2022) note that 3 of the papers analyzed failed to either provide enough details or the results reported were "not comprehensive". This issue then compounds as a sizable group of papers would leave out important information such as the specifications of how they created their private dataset, and other key details needed to replicate their findings. This information is critical for understanding how efficient each method is. This can be remedied by having a standard for reporting the results of AI/ML for phishing detection.

A solution would be to standardize what results are included in studies. This standard should require: a) the explicit location and name of which dataset was used, b) the algorithm used, C) explicit instructions on how the model was trained, D) an in-depth breakdown of false positives and negatives and true positives and negatives, and E) analysis execution speed.

Going forward there appears to be two paths when it comes to designing a defensive tool against fraudulent websites. The first approach would be focused on designing a client-focused service that would run a classifier on the hardware provided. The second approach would be to focus upon designing a cloud-based solution called through an API to offload the compute intensive work. Both of these approaches have their own benefits and drawbacks, which will be discussed in greater detail

in the next section, but either are a good beginning step for advancing anti-phishing measures.

3.3. Example of a Client-Based Solution

The following is a proposed framework for a Client-based solution for an anti-phishing extension. The solution should be built in a browser native language, such as JavaScript, using the provided machine learning libraries such as TensorFlow. When the website is accessed the extension will check a maintained whitelist which contains commonly used and trusted websites such as search engines, online office tools, and other trusted websites. Then, if the website is not trusted the extension will harvest data needed for classification on the model used. For this example it will be assumed that an ensemble classifier such as Random Forest will be used. The classifier will account for multiple features including domain information, the URL, and content on the website itself. Something similar to the feature set suggested by Mandadi et al. (2022), which lists DNS Record, Website Traffic, Age of Domain, End Period of Domain, IFrame Redirection, Status Bar Customization, Disabling Right Click, Website Forwarding, Domain, IP Address, "@" Symbol, Length, Depth, Redirection "//", "HTTP/HTTPS" in Domain name, Using URL Shortening Services "Tiny URL", Prefix or Suffix "-" in Domain (Mandadi et al., 2022).

The extension should have a pre-built model based upon the above implemented in the extension, with updates to reflect trends in current phishing websites. While the extension classifies the website the extension should have an interim page that will update when classification is done to either send the user to the website or inform the user of the detected security risk.

This model is considerably easy to implement and can theoretically be run on most modern workstations. This model also can be updated when performance drops due to changing trends in phishing to counteract the loss in accuracy, however doing so would require a consistent team to continuously watch the current trends in phishing websites. Another weakness of this model is the potential for False Positives and other accuracy issues, which would slow down the average user's speed of use. The proposed model will also need to determine if the link is safe or unsafe rapidly, else earning the ire of the end user.

These factors would need to be mitigated for a commercial implementation, by either optimizing the classification process, designing unique methods to obfuscate the methods in an unnoticed way, or other similar ideas (Fig. 1).

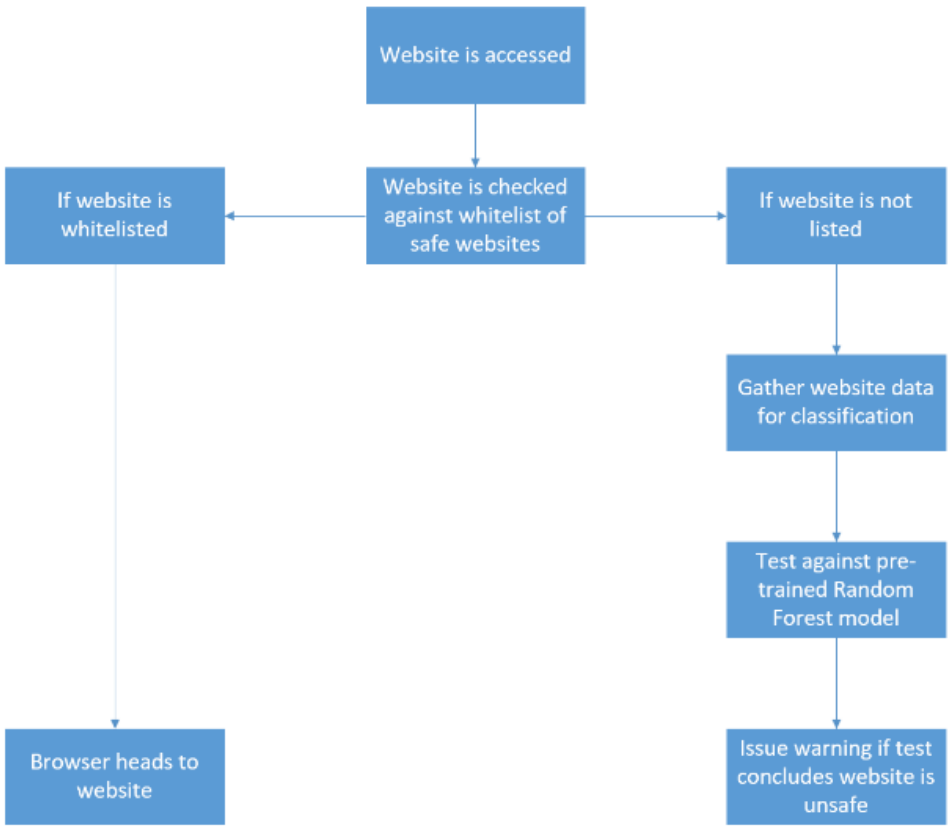

Fig. 1 A diagram of a simple client-based anti-phishing solution

3.4. Example of a Cloud-Based Solution

Where the prior solution is relatively simple to implement, the following is much more difficult due to the necessity of powerful computational processes, which are then hosted on either a public cloud service or a private cloud. This version would use a deep learning method, such as CNN-LTSM, which would be trained using information from repositories such as PhishTank. The classifier should be guided to look at meta information, website content, and the website URL itself. This trained model will then act upon information sent to it from client devices and determine if the site is a phishing website or a legitimate website. The model will then add that information into the next training set to continuously update the dataset to have it evolve naturally to counter new methods of Phishing as they appear as suggested by Sanchez-Paniagua et al. (2022).

This model, while simple to outline, is difficult to execute for practical use. For effective deep learning data needs to be consistently fed to the model for it to stay up-to-date. Supporting this infrastructure would cost a lot of money or resources to execute effectively, alongside the customization needed to optimize the classification processes. Ignoring those issues, another issue that one will run into is ensuring uptime for those dependent on the software. The cloud focused model requires consistent back and forth between all users and the classification service at all times for effective use. This also will require a lot of resources to implement. Once the model is properly trained and maintained, it however has the potential for a higher accuracy than its ensemble based brother

above. In the deep learning studies surveyed for this paper, most reported an accuracy of 97% or more, trumping the average next highest classifier which was often the Random Forest algorithm. Therefore, there is potential for cloud-based anti-phishing techniques powered by machine learning and artificial intelligence but the resource cost will limit effective implementation without serious capital investment (Fig. 2).

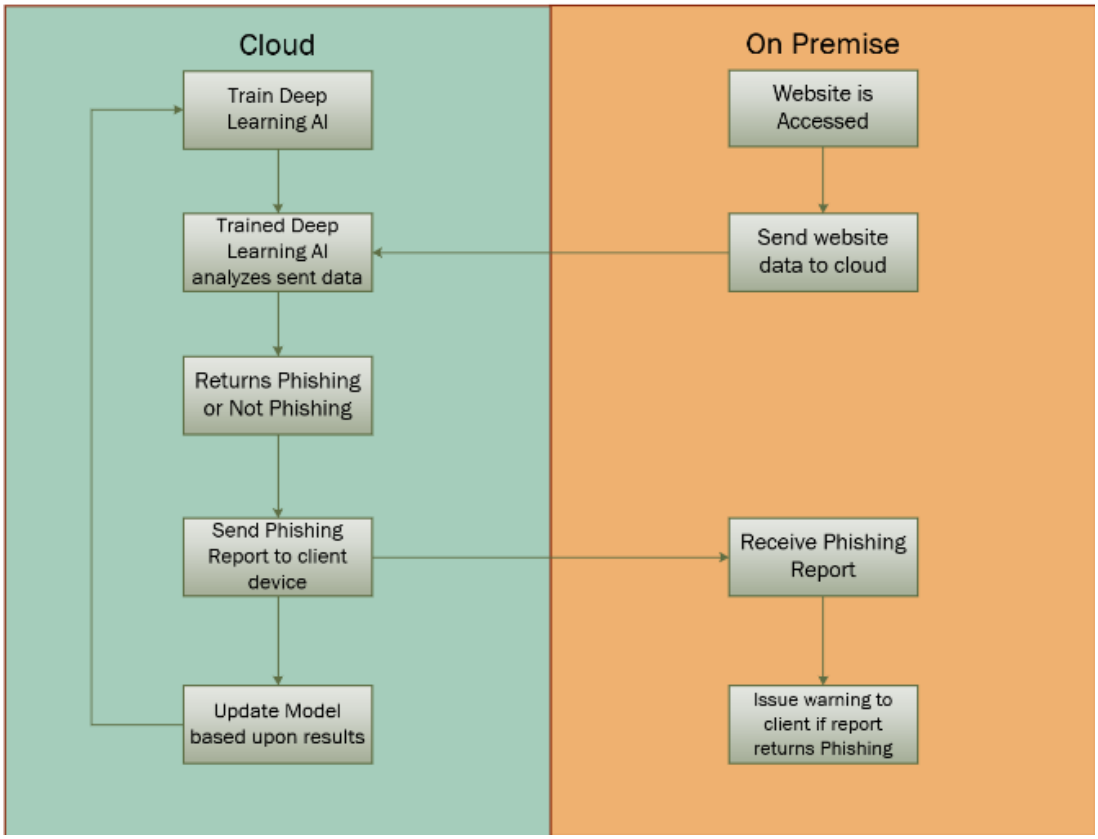

Fig. 2 A diagram of a simple cloud-based anti-phishing solution

3.5. Future Work

A prudent first step would be to standardize reporting of machine learning and artificial intelligence. Currently there is no codified standard for reporting Machine Learning and Artificial Intelligence study results. Some studies contain everything needed to replicate the experiments performed and how the conclusion was drawn; however other studies will leave out needed details for conclusive analysis or replication. Mourtaji et al. (2017) for example outlines their own framework and show results from said framework without supplying the dataset used in testing, which they claim to have pulled from PhishTank and Alexa to populate. By providing the dataset used in testing to an online repository for verification it allows for doubt to be cleared and will be of great assistance to other researchers in the field.

By focusing on standards that ensure easy replication of results, and clarity

within the information reported, other researchers will be able to work off of the research and develop the new technologies. Therefore we would like to suggest a framework that would include these specifications for all reported testing: a repository containing the training dataset and testing dataset used, the features selected for classification, the classifier used alongside documentation of how to implement custom classifiers, the true positives and negatives alongside the false positives and negatives from resulting validation tests, precision rating, recall rating, accuracy rating, and F1 score. Alongside this information there should be enough instruction for the reader to validate the paper by replicating the experiment within. By including this information it shall ensure reliable replication, which will make it easier to build upon thus helping the proliferation of information.

On a more practical level the next step should be creating working models and testing them in live environments. By making a model, Client or Cloud based, will allow for researchers to see the practical shortcomings to these methods and correct them. Once the shortcomings are known more development can take place evolving the field, which will help combat one of the most common threats on the internet.

4. CONCLUSION

Phishing is one of the most common threats to cybersecurity in the current world. Many organizations have become acutely aware of the potential danger of a successful attack. This has led to an increased focus on developing new technologies to prevent such attacks from taking place. By using machine learning and artificial intelligence many posit a learning defensive system that can prevent website phishing attacks and lower potential vectors for attack.

Currently there is no cure-all with many papers acknowledging the ever-changing nature of website based phishing attacks, preventing a permanent solution. However, a well automated system could go a long way to preventing website-based phishing attacks and could be a useful solution for major organizations. Most studies believe that a web extension for modern web browsers such as Google Chrome is where companies should look for future developments. A development of a working model for testing in live environments would do well in advancing the field by showing what potential shortcomings exist.

Finally, there is a lack of standardization in the reporting of data done in the multitude of studies focusing on the topic. To better advance the field in the focus of implementing anti-phishing ML/AI into working prototypes, a standard of reporting would make it easier to gather information. By always including the dataset used, the algorithm used, the instructions for training the model, a breakdown of the training and testing results and a record of time taken to execute a task, it would allow for information to be disseminated and processed faster which in turn could assist in the development of such anti-phishing technologies.

REFERENCES

Abdelnabi, S., Krombholz, K., & Fritz, M. (2020, October). VisualPhishNet: Zero-day phishing website detection by visual similarity. In *Proceedings of the 2020 ACM SIGSAC conference on computer and communications security* (1681-1698).

Ali, W., & Malebary, S. (2020). Particle swarm optimization-based feature weighting for improving intelligent phishing website detection. *IEEE Access*, *8*, 116766-116780.

Alsariera, Y. A., Elijah, A. V., & Balogun, A. O. (2020). Phishing Website Detection: Forest by Penalizing Attributes Algorithm and Its Enhanced Variations. *Arabian Journal for Science and Engineering, 45*(12), 10459–10470.

Assefa, A., & Katarya, R. (2022, March). Intelligent Phishing Website Detection Using Deep Learning. In *2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS)* 1, 1741-1745. IEEE.

Chen, J. L., Ma, Y. W., & Huang, K. L. (2020). Intelligent Visual Similarity-Based Phishing Websites Detection. *Symmetry*, *12*(10), 1681.

Mandadi, A., Boppana, S., Ravella, V., & Kavitha, R. (2022, April). Phishing Website Detection Using Machine Learning. In *2022 IEEE 7th International conference for Convergence in Technology (I2CT)* (1-4). IEEE.

Mourtaji, Y., & Bouhorma, M. (2017, October). Perception of a new framework for detecting phishing web pages. In *Proceedings of the Mediterranean Symposium on Smart City Application* (1-6).

Sánchez-Paniagua, M., Fernández, E. F., Alegre, E., Al-Nabki, W., & González-Castro, V. (2022). Phishing URL Detection: A Real-Case Scenario Through Login URLs. *IEEE Access*, *10*, 42949-42960.

Saravanan, P., & Subramanian, S. (2020). A framework for detecting phishing websites using GA based feature selection and ARTMAP based website classification. *Procedia Computer Science*, *171*, 1083-1092.

Subasi, A., & Kremic, E. (2020). Comparison of adaboost with multiboosting for phishing website detection. *Procedia Computer Science*, *168*, 272-278.

Suleman, M. T., & Awan, S. M. (2019). Optimization of URL-based phishing websites detection through genetic algorithms. *Automatic Control and Computer Sciences*, *53*(4), 333-341.

Zhou, J., Liu, Y., Xia, J., Wang, Z., & Arik, S. (2020). Resilient fault-tolerant anti-synchronization for stochastic delayed reaction–diffusion neural networks with semi-Markov jump parameters. *Neural Networks*, *125*, 194-204.

Zhou, Z., & Zhang, C. (2022, May). Phishing website identification based on double weight random forest. In *2022 3rd International Conference on Computer Vision, Image and Deep Learning & International Conference on Computer Engineering and Applications (CVIDL & ICCEA)* (263-266). IEEE.