Understanding the Impact of Bad Words on Email Management through Adversarial Machine Learning

Jonathan He
Princess Anne High School, USA
jonathanhe12345678@gmail.com

Qi Cheng
Johns Hopkins University, USA
qcheng7@jhu.edu

Xiangyang Li
Johns Hopkins University, USA
xyli@jhu.edu

ABSTRACT

Information security teams' ability to handle essential email services depends on their accumulated knowledge and experience of legitimate and spam messages received in the past. Numerous organizations have stored huge datasets of legitimate and spam messages over the years, and they could leverage these data to improve their spam detection knowledge and capabilities. As a powerful tool, spam filters that employ machine learning models have been instrumental in fighting back against spam and phishing emails. Recent studies have shown that these filters are susceptible to adversarial evasion attacks that can exploit these models by purposely modifying malicious emails to bypass detection, such as the popular good word attacks. However, not enough attention has been paid to the issue of normal emails that cause, in a similar manner, false positives in these spam filters. This risk may rise as these misclassifications follow a different threat model. Computer users accidentally, or deliberately in some uncommon scenarios, write business emails that include certain words, coined "bad words" in this paper, that result in the emails being flagged by a spam filter. This paper reports an effort of developing a novel method to identify potential bad words and to analyze their impact. Using two different datasets, preliminary experimentation results have spotlighted real concern about such words' ability to cause interruptions to normal email communications. These results and their generated knowledge about bad words can help information security and messaging teams develop more effective spam detection tools.

KEYWORDS

Email management; Spam detection; Bad words; Adversarial Machine Learning; False positive

INTRODUCTION

Email is still the most important messaging service for personal and business communications, despite the rapid development of other platforms. The necessity of securing emails has generated applications in spam filters to employ machine learning (ML) models and additional functionalities, such as integrated enterprise directory queries. Most adversarial attacks efforts focus on misclassifying spam emails. Extensive studies have focused on reducing the occurrence of false negatives regarding undetected spam and phishing emails before they reach an end-user. However, false positives of these spam filters, i.e., normal emails being flagged, cause communications to be dropped or blocked. Missing legitimate emails can even lead to worse consequences than receiving spam emails, for many users (Christina, Karpagavalli, & Suganya, 2010).

This study makes several contributions to the field of messaging management. We examine false positives by ML-based spam filters that flag normal business emails, to better understand their causes and remediations. More importantly, this paper describes a novel approach that takes advantage of adversarial ML perturbations to model input (the feature space) to identify bad words in emails (the problem space) that can cause false positives. This provides a new perspective on other relevant issues and implications.

ATTACKS ON SPAM FILTERS

ML classification models, such as Support Vector Machine (SVM) models (Olatunji, 2017) or Bayesian classification models (Sahami et al., 1998), have been employed for spam email detection. In these models, a set of features are extracted, based on the words and phrases in an email, and these features are input to the models to classify this email as a normal email or a spam email.

Adversarial Attacks

Wittel and Wu (2004) categorized adversarial attacks on spam email detectors into three types: *tokenization attacks*, in which spammers intend to disturb the tokenization of the email content by splitting or modifying features; *obfuscation attacks*, in which the email content is obscured from the detector using encoding or misdirection; and *statistical attacks*, in which spammers attempt to skew the message's statistics to distract the detector. An example of

a popular statistical attack is the so-called "good word attack" (Lowd & Meek, 2005), in which a set of words were identified and added to emails against two types of statistical spam detectors: maximum entropy and naive Bayes filters. However, most of these approaches basically search among the variants of an email by substituting or deleting words for NLP adversarial attack examples to cause a desired change in classification (Yoo et al., 2020).

Adversarial Machine Learning

Recently, we have seen adversarial machine learning attacks that perturb model input features, so the classification output is changed as desired. Such attacks on machine learning models were proposed in domains including image classification (Goodfellow et al., 2014) and Voice Processing Systems (Carlini & Wagner, 2018). They applied algorithms such as Projected Gradient Descent (PGD) (Madry et al., 2017) and Carlini and Wagner Attacks (CW) (Carlini & Wagner, 2017) against deep-learning models. Less attention is given to computer security applications that process other types of data, such as the text content of an email used by spam filters. Moreover, such attacks perturb the features; the translation of an adversarial attack in the feature space back to a realistic event in the problem space, such as an email, is critical to completing a realistic attack process.

A recent work developed an efficient method to identify good or "magic" words based on changes in the feature space that cause spam emails to bypass spam filter models (Wang et al, 2021). This is novel to use findings of feature perturbations to guide the changes needed in modifying emails. This paper extended this method to study the "bad" words, which can cause normal emails to be flagged, resulting in loss of productivity.

APPROACH

A Term Frequency-Inverse Document Frequency (TF-IDF) vector was calculated for every email. Then we trained an SVM classifier with these TF-IDF features as input. PGD was run on the trained SVM classifier to perturb a set of randomly selected legitimate emails, which created a set of adversarial perturbations to their TF-IDF vectors. Based on these changes, we identified a set of bad words that could be added to normal emails to trigger false positives by the SVM spam filter.

Adversarial Perturbations in the Feature Space

Calculating TF-IDF from the words appearing in an email is a common method to vectorize textual information into numeric values:

$$TF_{i,j} = \frac{N_{i,j}}{\sum_k N_{k,j}}$$

$$IDF_i = \log \frac{|D| + 1}{|j : t_i \in d_j| + 1} + 1$$

where Ni,j is the number of times word ti appears in email dj; |D| is the total number of emails in the corpus; |j : ti ∈ dj| indicates the number of emails containing the term ti. The IDF term is smoothed for those common words appearing in every document. The higher frequency of the appearance of a word in a particular file and the lower file frequency of the word in the entire file collection results in a higher TF-IDF value, which reflects the significance of the word or feature used in the classification model.

An SVM classifier expands the original data dimensions to separate the samples in the transformed high-dimensional space (Chen, Lin, & Schölkopf, 2005). If the number of features is much larger than the number of samples, a linear kernel is recommended, to avoid over-fitting the SVM model. This is the case for our experimentation.

PGD is an iterative algorithm that finds the disturbance to features with a constraint, *dmax*, which is the Euclidean distance measuring how much change can be made to feature values to achieve the maximum loss in classification (Mardry et al., 2017). In our approach, PDG changed a set of randomly selected legitimate emails that each generated an adversarial example in a TF-IDF vector.

Finding Bad Words

The goal is to add special words, called "bad words", into ham emails to fool the classifier into misclassifying them as spam emails. These so-called bad words are found by intersecting two sets of words. The first set of words contains unique spam words that only appear in spam emails. The second set has 100 words that correspond to the top 100 features that were changed the most by the PDG algorithm. The ranking of these features is done by their variance of the feature value changes over all the selected legitimate emails. These top 100 features happen to be some words

added by PGD, i.e., TF-IDF features changed from zero to non-zero value during the perturbation process. These words are not too many, so adding them to a legitimate email should not change the nature of the email.

EXPERIMENTATION

Datasets

We used a dataset, called Lingspam, of 2,412 legitimate emails and 481 spam emails (available at https://www.kaggle.com/mandygu/lingspam-dataset) and another dataset, called Tutorial, of 4149 legitimate emails and 1889 spam emails (available at https://spamassassin.apache.org/old/publiccorpus/). Messages sent by the owner of the mailbox, all HTML tags, the headers of the messages, and spam messages written in non-Latin character sets were removed before analysis. To reduce both complexity and subsequent processing, we also removed all special symbols, numbers, and stop words. Additionally, we aggregated words through stemming, and we replaced the URL link in each email by the word "URL."

Experimental Settings

First, the emails in each dataset were randomly divided into a training set and a test set, according to the ratio of 4:1. We used the Sklearn library (https://scikit-learn.org/) in processing the emails. We trained the SVM classifier in the SecML library (https://secml.gitlab.io/). The default penalty factor was selected when calling the method "best_estimate()" in training the classifier. In testing, the SVM classifier achieved 99.15% accuracy on the Lingspam dataset (a 0.34% false positive rate and a 0.51% false negative rate) and 98.27% accuracy on the Tutorial dataset (a 0.55% false positive rate and a 1.18% false negative rate).

Next, the projected gradient descent (PGD) algorithm in the SecML library was used for adversarial perturbations. Generally, larger *dmax* settings are likely to lead to a greater success rate of inducing misclassification. So, we chose a variety of different *dmax* settings, ranging from 0.02 to 0.5. From the test dataset, we randomly selected different numbers of legitimate emails for the PGD algorithm to work on.

Finally, we added the identified bad words to the original legitimate emails and re-calculated the TF-IDF vectors from these modified emails to examine the success rate of these emails to incur false positives by the spam filter.

Results

The experimentation focused on two parameters: the number of emails being used and the *dmax* setting in PGD perturbations. Here, we examine four types of results that show the potential problems of interest to us: the success rate of PGD perturbation of a feature input, the success rate of adding bad words causing the ML to misclassify an otherwise legitimate email, the set of bad words being identified, and the size of the bad word set.

As in Table 1, several tests were performed on the Lingspam dataset by varying the number of emails used by PGD perturbation while keeping *dmax* in PGD perturbation the same. After analyzing the results, we saw that there was no clear relationship between the number of emails used by PGD perturbation and the success rates of PGD and bad success rates.

Number of Emails Used by PGD Perturbation	Dmax Used in PGD Perturbation	PGD Perturbation Success Rate	Bad Word Success Rate	Bad Word Count
100	0.15	0.89	0.4237147595	17
200	0.15	0.95	0.4237147595	17
481	0.15	0.9272349272	0.4237147595	17

Table 1. Results for the different numbers of emails used in PGD perturbation on the Lingspam dataset

Next, we varied *dmax* in the PGD perturbation while the number of emails used by PGD perturbation was kept constant at 200. As shown in Figure 1, we noticed that, as *dmax* increases, the success rate of perturbation increases as well; this was expected. However, the success rate of bad words to fool the SVM detector did not always go up; this showed a more complicated relationship between these two measures. But there was a positive correlation between the number of bad words and their capability to cause false positives.

Figure 1. Results showing the number of bad words identified on the Lingspam dataset and their corresponding success rates of PGD perturbation and bad words

Figure 2 shows the results for the Tutorial dataset with similar patterns. However, it shows these trends were not the same as those of the Lingspam dataset.

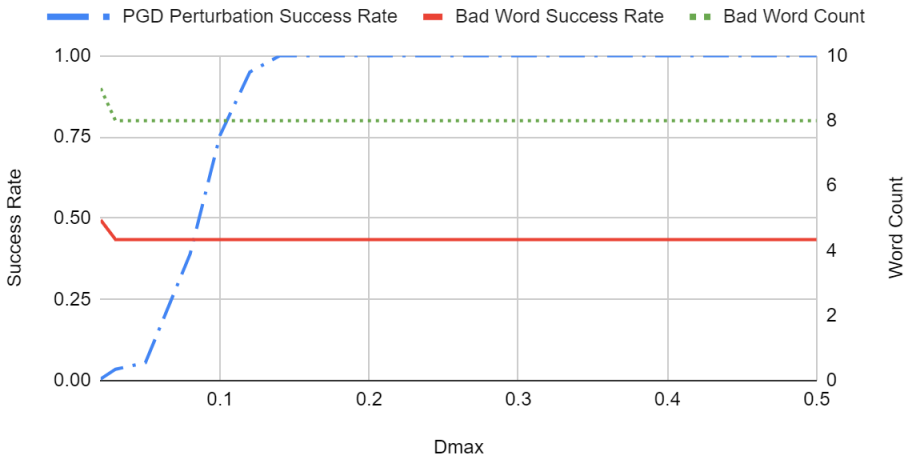

Figure 2. Results showing the number of bad words identified on the Tutorial dataset and their corresponding success rates of PGD perturbation and bad words

After analyzing the performance results, a case study was conducted on several emails that was intended to show how bad words can change individual emails. It would further examine whether fewer bad words can achieve the effect of turning a ham email into a spam email. This case study was on the Lingspam dataset, and ten ham emails were selected. These ten emails were further tested, and they all produced an output of 0, verifying that they were ham emails to begin with. We chose to use 200 emails used by the PGD perturbation and *dmax* at 0.14 as these settings produced one of the highest success rates to cause false positives by the identified bad words.

The set of 18 bad words includes these: {*psoriasis guild prospectus listbot ffa recipe nbsp ambra signups nek plurabelle onlinenow infoseek wspn parasite hollander mhz babez*}. Next, we added bad words, one by one, to the original email until the email is classified as a spam email. The results showed that five were classified as spam after adding all 18 words in the bad word set. However, just adding just seven bad words could cause the ML model to flag

one of the five emails (see Table 2). This demonstrated how easy it is for a ML-based spam filter to generate a false-positive error.

Original Email	Modified Email
i ' m about to computerize a sociolinguistic corpus of spoken french and english . i would be grateful to anybody who could give me some information about concording and text retrival softwares . i ' m using a macintosh micro-computer . thanks . . . louise charbonneau - lloyd	i ' m about to nbsp computerize a sociolinguistic corpus of spoken prospectus french and english . i would be grateful to listbot anybody who could give me some ffa information about recipe concording and text retrival softwares guild . i ' m using a macintosh psoriasis micro-computer . thanks . . . louise charbonneau - lloyd

Table 2. Normal email and the modified email with bad words

DISCUSSION

Susceptibility of Machine-Learning Based Spam Filters to Bad Words

The results show the false positive issues related to the ML-based spam filters, which misclassify normal business emails as spam emails. Only seven bad words being added to a legitimate email led to false positives. This not only reveals the challenges to the classification accuracy with ML models but also exposes the vulnerabilities of these models when adversarial ML attacks occur. Our experiment provides a new perspective to explain why ML-based spam filters may suffer from bad word mischief that can lead to classification errors. The inclusion of some bad word could accidentally cause unintended consequences to other machine learning-based classification or clustering tasks. Defense mechanisms need to be further developed, to protect ML algorithms and to reduce the implications of adversarial ML attacks. For example, ML models need to be re-enhanced through learning that is informed by these bad words.

Implications for Email and Messaging Management

There have been frequent complaints that important emails that people need to see are filtered out into their spam folders. Emails that are accidentally marked as spam, and hence not delivered to a user's main email list, can lead to severe consequences. Organizations need to leverage the knowledge that they accumulate about legitimate and spam emails over time, and then construct their own data sets about spam emails so that they can build more customized, effective, and robust spam filtering solutions in terms of minimizing the email false positive problem, which is one of the key challenges in spam filtering (Islam et al., 2008). In addition, employees need to be educated to become aware of email's false positive problem so that they can recognize what kinds of bad words may contribute to the email false positive problem and develop a habit to check the spam folder regularly so that important emails will not be missed.

CONCLUSION

This paper reports a novel method to identify potential bad words and analyze their impact. Our results demonstrate that ML-based spam filters may suffer from bad word attacks to misclassify legitimate emails as spam emails. These results can help develop more effective spam detection tools.

Currently, there are no spam filtering solutions that can claim 0% false positives and 0% false negatives. Thus, spam filtering solution developers need to continuously refine and improve the algorithms or techniques used by their spam filters to improve their accuracy and to eliminate the email's false positive and false negative problems. Further research is needed to develop more accurate techniques and to identify how to better prevent the effect of bad words in reducing instances of email's false positive problem.

REFERENCES

Carlini, N., & Wagner, D. (2017). Towards evaluating the robustness of neural networks. *In the proceedings of 2017 IEEE Symposium on Security and Privacy (SP)*, pp. 39–57.

Carlini, N., & Wagner, D. (2018) Audio adversarial examples: Targeted attacks on speech-to-text. *In the proceedings of 2018 IEEE Security and Privacy Workshops (SPW)*, pp. 1–7.

Chen, P., Lin, C., & Schölkopf, B. (2005). A tutorial on -support vector machines. *Applied Stochastic Models in Business and Industry*, 21(2), 111–13.

Christina, V., Karpagavalli, S., & Suganya, G. (2010). A study on email spam filtering techniques. *International Journal of Computer Applications*, 12(1), 0975-8887.

Goodfellow, I. J., Shlens, J., & Szegedy, C. (2014). Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572.

Islam, R., Singh, J., Chonka, A., & Zhou, W. (2008). Multi-classifier classification of spam email on a ubiquitous multi-core architecture. In *2008 IFIP International Conference on Network and Parallel Computing*, pp. 210-217.

Lowd, D., & Meek, C. (2005). Good Word Attacks on Statistical Spam Filters. In *CEAS* (Vol. 2005).

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv*:1706.06083.

Olatunji, S. O. (2017). Extreme Learning machines and Support Vector Machines models for email spam detection. In *2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE)* (pp. 1-6). IEEE.

Sahami, M., Dumais, S., Heckerman, D., & Horvitz, E. (1998, July). A Bayesian approach to filtering junk e-mail. In *Learning for Text Categorization: Papers from the 1998 workshop*, Vol. 62, pp. 98-105.

Wang, C., Zhang, D., Huang, S., Li, X., & Ding, L. (2021, May). Crafting Adversarial Email Content against Machine Learning Based Spam Email Detection. In *Proceedings of the 2021 International Symposium on Advanced Security on Software and Systems*, pp. 23-28.

Wittel, G. L., & Wu, S. F. (2004). *On Attacking Statistical Spam Filters*. In CEAS.

Yoo, J. Y., Morris, J. X., Lifland, E., & Qi, Y. (2020). *Searching for a search method: Benchmarking search algorithms for generating NLP adversarial examples*. arXiv preprint arXiv:2009.06368.