Adversarial Machine Learning for Spam Filters

Bhargav Kuchipudi, Ravi Teja Nannapaneni and Qi Liao*
Department of Computer Science
Central Michigan University
Mt. Pleasant, Michigan, USA

ABSTRACT

Email spam filters based on machine learning techniques are widely deployed in today's organizations. As our society relies more on artificial intelligence (AI), the security of AI, especially the machine learning algorithms, becomes increasingly important and remains largely untested. Adversarial machine learning, on the other hand, attempts to defeat machine learning models through malicious input. In this paper, we experiment how adversarial scenario may impact the security of machine learning based mechanisms such as email spam filters. Using natural language processing (NLP) and Baysian model as an example, we developed and tested three invasive techniques, i.e., synonym replacement, ham word injection and spam word spacing. Our adversarial examples and results suggest that these techniques are effective in fooling the machine learning models. The study calls for more research on understanding and safeguarding machine learning based security mechanisms in the presence of adversaries.

CCS CONCEPTS

• **Information systems → Spam detection**; • **Theory of computation → Adversarial learning**.

KEYWORDS

Network security, spam detection, adversarial machine learning, artificial intelligence

ACM Reference Format:
Bhargav Kuchipudi, Ravi Teja Nannapaneni and Qi Liao. 2020. Adversarial Machine Learning for Spam Filters. In *IWCC '20: 9th International Workshop on Cyber Crime, August 25–28, 2020, Dublin, Ireland.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Electronic mail (email) is an intrinsic part of our daily life for the past decades. Therefore, email security becomes extremely important for the overall health of Internet. Unsolicited messages, or email spam, have become a hard problem since the invention of email. According to Message Labs Intelligence Report [13], spam now comprises approximately 88% of all email traffic. While there are much research on fighting the spam problem, the most common approach adopted by organizations is to deploy email spam filtering technologies that utilizes either known signatures or more recently machine learning based approaches [3].

With the rapid advance of artificial intelligence (AI) and widespread machine learning applications, researchers become cautious of the security of such AI and its trustworthiness. Recently, adversarial machine learning emerges as a technique that attempts to misguide machine learning models through malicious input. While difficult, researchers have successfully identified adversarial examples to bypass classifiers [6]. If such techniques are successful, the vast majority of machine learning based security mechanisms will be at risk since the decision making by those models may be compromised and no longer trustworthy.

In this paper, we experiment adversarial machine learning on machine learning based anti-spam technologies. Can a machine learning classifier such as an email spam filter be manipulated by attackers? How may we invade it, and eventually, improve it so that it can be resistant to attacks? Since Bayesian models have been proved to be an effective way to fight email spams [1, 5, 7, 11, 15, 16] and widely adopted, we use a Naive Bayesian classifier as an example to study the effect of adversarial learning on spam filtering. In particular, we implement three techniques to invade the spam filter, i.e., synonym replacement, ham word injection, and spam word spacing.

In all the above techniques, we are able to preserve the original meaning of the messages after replacing, injecting or spacing words. To trick the spam filter to misclassify emails, we found numerous adversarial examples that make spam be classified as ham and go through the spam filter. Conversely, it may also be possible to have ham classified as spam so legitimate emails are dropped. Our findings suggest it is possible for adversaries to utilize adversarial machine learning to destabilize spam filters. The study serves to emphasize the importance of the security of AI/machine learning and its application in cybersecurity.

The rest of the paper is organized as follows. Section 2 reviews related literature on spam filters. Section 3 formalizes the machine learning models for spam detection and discusses the key invasion techniques. Following implementation details in Section 4, we present the experimental results and key findings of the study. Finally, Section 5 concludes our work.

2 RELATED WORK

Naive Bayesian model is the most popular statistical-based anti-spam method for its strong categorization and high precision [5]. Studies have shown that Naive Bayes Classifier has been proved to be effective in practice [15]. In particular, tradeoffs of five different versions of Naive Bayes for spam filtering have been studied [7].

*Corresponding author: liao1q@cmich.edu

FARES '20, August 25–28, 2020, Dublin, Ireland
© 2020 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

Over time, researchers have proposed hybrid approaches. For example, hybrid Bayesian classifier approaches [11] were proposed using local and global classifiers to detect spam. Using both association rule and Naive Bayesian Classifiers was recommended [16]. Combining Naive Bayesian classifier and an alternative memory-based approach may achieve more accurate spam filtering [1]. A new spam filtering method [5] based on Naive Bayes and biologically inspired artificial immune system (AIS) was also proposed, and the result shows that the hybrid algorithm is more robust.

Besides Bayesian model, other machine learning techniques have been studied for spam detection. An adaptive statistical data compression model [2] was proposed for spam filtering. Messages were modeled as sequences and probabilistic text classifiers were developed based on character-level or binary sequences. Studies have evaluated the accuracies of spam classifiers such as support vector machine (SVM) with a linear kernel, a logistic regression (LR) classifier, and Multiple Instance Logistic Regression (MILR) [9]. The utility of over 40 features of email was investigated by calculating information gain of these features over ham, spam and phishing corpora [13]. In addition, the effectiveness of collaborative spam filters [14] has been studied. In contrast to training with own email in each individual organization, spam filters may be trained with large corpora data from a variety of sources including legitimate and spam messages from many sources to many destinations.

However, most of the proposed machine learning approaches do not consider the presence of adversaries that may launch sophisticated attacks to undermine deployed spam detectors either during the training or the prediction phase. Despite the success of above algorithms in detecting spam, the presence of adversaries undermines the performance of spam filters [3]. Adversarial machine learning [6] exposes the vulnerabilities of machine learning based security mechanisms. This technique is also known as reverse engineering in machine learning. Modern machine learning models can be broken in different ways shown by different adversarial examples. Recent survey reveals a new type of spam tweet (adversarial spam tweet) that can attack against online social networks such as Twitter spam detectors [3]. Study shows spam content may be mixed with legitimate content to create the camouflaged messages [14]. With the knowledge of email distribution, the attacker can select a smaller dictionary of high-value features that are still effective [10].

Adversarial machine learning, however, has limitations. The adversary's level of knowledge about the deployed model plays an important role in determining the success of attacks. The adversary may know the machine learning algorithms used, or the importance of features used by the deployed model. The amount of information can be limited. For example, researches have shown that an adversary can exploit statistical machine learning in spam filters even if the attacker has access to only 1% of the training dataset [8]. The effect of dictionary based attacks and well-informed focused attacks, however, may be reduced by adding weights for classifiers [10].

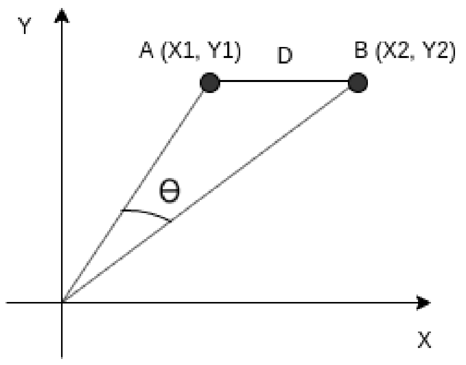


Figure 1: Email similarity may be calculated using a cosine similarity metric between two email vectors A and B.

3 SPAM INVASION METHODOLOGY

In this section, we discuss the Naive Bayesian model as an email spam filter and the three invasion techniques, i.e., synonym replacement, ham word injection and spam word spacing. An email similarity metric is used to preserve the meaning of original messages. The algorithm of constructing new messages from the original messages using the invasion techniques is presented.

3.1 Naive Bayesian Model

From Bayes' theorem and the theorem of total probability, the conditional probability ($P(S|W)$) that an email is spam given that the email contains word W is as follows:

$$P(S|W) = \frac{P(W|S) * P(S)}{P(W|S) * P(S) + P(W|H) * P(H)} \tag{1}$$

where S represents the spam email, H represents the ham email, and $P(W|S)$ and $P(W|H)$ are the conditional probabilities of the word W.

First, an email is tokenized. Second, the tokens are converted to a matrix of token counts. Third, the count matrix is transformed into a normalized representation using Term Frequency (tf) and Term Frequency - Inverse Document Frequency (tf-idf), i.e., *tf-idf(t,d) = tf(t,d)* $*$ *idf(t)*, where $idf(t) = \log[\frac{n}{df(t)}] + 1$ and *df(t)* is the document frequency of term t. Using tf-idf instead of the raw frequencies can scale down the impact of tokens that occur very frequently. The intuition is that if a word appears frequently in an email, it should be important and we should give that word a high score. On the other hand, if a word appears in many other documents, it is probably not a unique identifier, therefore we should assign a lower score to that word. While there are multiple distributions of Naive Bayes models such as Gaussian, multinomial, or Bernoulli, we choose to use multinomial Naive Bayes model since we are dealing with discrete features such as word counts.

3.2 Email Similarity

Our primary goal is to modify an email message (e.g., spam) M_0 such that the modified sample M^* can both satisfy the needs (e.g., does not change the nature of spam) and bypass the spam classifier.

In other words, the new messages must be similar to the original messages. We compute a similarity score between the new message and the original message using cosine similarity (Figure 1). Mathematically, cosine similarity measures the cosine of the angle between two vectors projected in a multi-dimensional space. The smaller the angle, the higher the cosine similarity.

Algorithm 1 Construction of new messages with synonym replacement.

```
Input: AM(actual message), SD(synonym dictionary for all words in the actual message), and R(Range, largest synonym set in the dictionary)
Output: new message
procedure CONSTRUCTNEWMESSAGES($AM$, $SD$, $R$)
    $new\_messages = []$
    $words = SD.keys()$
    $original\_msg = AM$
    $new\_msg = []$
    for each integer $i$ in $R$ do
        $new\_msg = new\_msg + original\_msg$
        for each word $i$ in $words$ do
            $syn\_arr = syn\_dict[word]$
            if $len(syn\_arr) > 0$ then
                if $len(syn\_arr) > i$ then
                    $new\_msg[new\_msg.index(word)] = syn\_arr[i]$
                else
                    $new\_msg[new\_msg.index(word)] = syn\_arr[-1]$
                end if
            end if
            $new\_messages.append(''.join(new\_msg))$
            $new\_msg = []$
        end for
    end for
    Return $new\_messages$
end procedure
```

3.3 Synonym Replacement

Intuitively, since the Naive Bayesian classifier utilizes the term frequencies, manipulating words in emails has a significant impact in spam classification. To increase the tendency of spam being classified as ham without changing too much its original meaning, we employ a synonym replacement technique based on natural language processing (NLP).

For a given word W, find a set of synonyms $S = \{W_1, W_2, W_3, ..., W_n\}$. A synonym word W^* is chosen from the set S to replace the word W. Replacing all the prominent words in an email with their closest synonyms to form a new message will deliver a similar meaning. The choice of synonyms depends on the similarity metrics discussed in the previous section. Since stop words (common words), such as "as", "the", and "it", do not play much role in changing the meaning of a message and have little effect on classification, we exclude stop words from synonym replacement.

3.4 Ham Word Injection

Since one of the key features on which the spam filter build is tf-idf, manipulating occurrence frequency of words is also promising. Assuming there is a publicly available database of the spam words that have high probabilities to trigger a spam filter, any words that are not included in the database are considered ham words. One may inject ham words in the messages at different places without changing much meaning of the original messages. When a message has enough ham words and reaches the tipping point, the model may classify a spam message as a ham message.

3.5 Spam Word Spacing

Spacing words is another interesting approach that may invade a spam filter. In this method we add spaces between the characters of words that may have high probability of being considered as spam words. Intuitively, by adding spaces between the characters, a text parser may consider each character as an individual word, thus disturbing word frequency distribution in the model.

4 EXPERIMENTAL RESULTS

In this section, we discuss a few implementation details as well as the general information about the dataset used in the experiments. First, the model performance is evaluated for accuracy without adversarial environment. Second, we present the adversarial examples using all three invasion techniques discussed in the previous section. Finally, we present the results and key findings of our study.

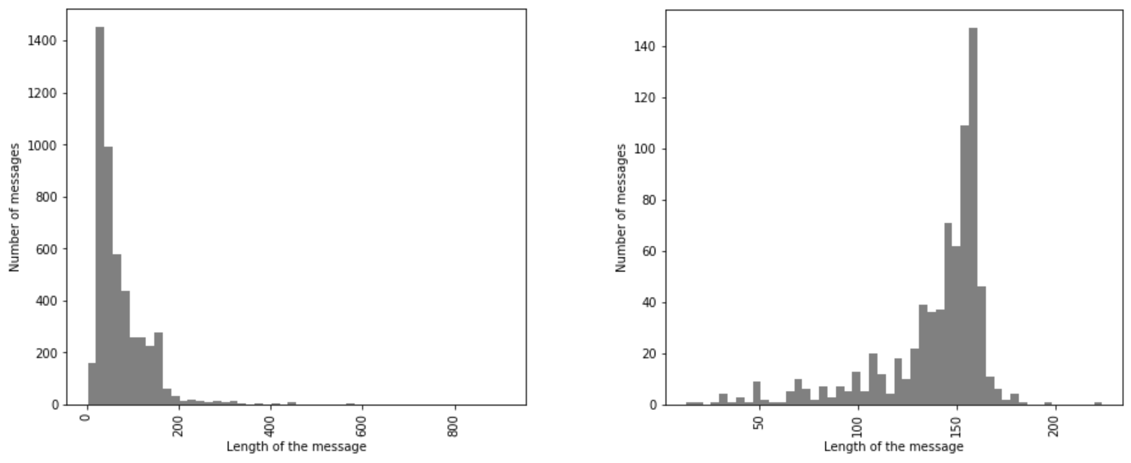
4.1 Implementations

The first step towards building a spam classifier is data pre-processing, which plays a key role in extracting the features and classifying an email as spam or ham. Pre-processing of the data contains several stages including tokenization and lemmatization.

A few `python` modules are used such as *csv*, *nltk*, *pandas*, *sklearn*, *countervectorizer*, *TfidfTransformer*, *MultinomialNB*, *classification_report*, *confusion_matrix*, etc. Data is read from *csv* files and tokenized by removing the stop words and punctuation that do not play a key role in predicting an email to be spam or ham. Multiple words with similar meaning are linked or grouped in the lemmatization process. The tokens are then converted to a frequency count matrix, which is normalized using tf and tf-idf. Based on this term frequency, we attempt to invade the spam filter using the techniques discussed in previous sections.

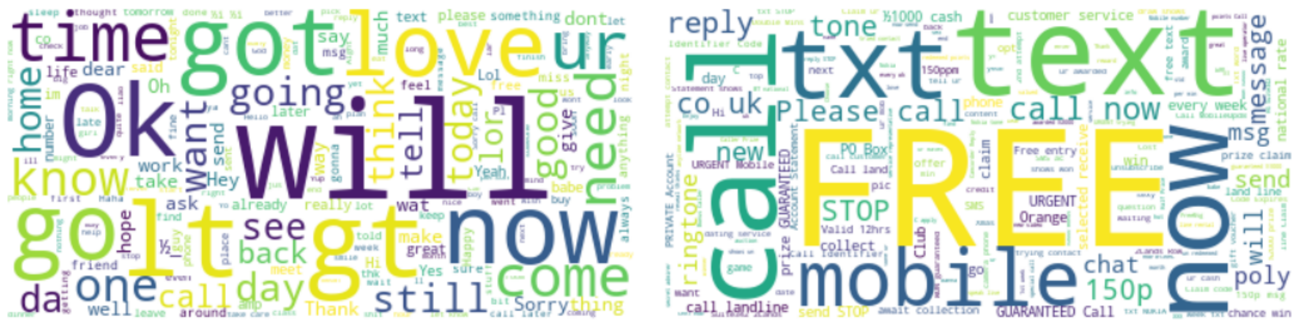
4.2 Dataset

We use a publicly available spam dataset on Kaggle [4], which contains a total of 5572 messages out of which 747 are spam and 4825 are ham messages. Each message is labeled with either ham or spam. An overview of message length distribution of the dataset is shown in Figure 2. It appears that most of spam messages are longer than ham messages with some exceptions. In addition, the top words in both ham and spam messages are illustrated in Figure 3.



(a) Ham messages (b) Spam messages

Figure 2: Length distributions: most legitimate messages are shorter than 150 characters while most spam messages center around 150 characters.



(a) Ham messages (b) Spam messages

Figure 3: Top words in messages excluding stop words.

4.3 Spam Filter Performance

Dataset is divided into separate training and test sets. To evaluate the performance of Naive Baysian spam filer, we use the standard precision and recall measurement. Precision is the ability of a classifier not to label an instance positive that is actually negative. For each class, it is defined as the ratio of true positives to the sum of true and false positives. Recall is the ability of a classifier to find all positive instances. For each class, it is defined as the ratio of true positives to the sum of true positives and false negatives. The F1 score is a measure of test accuracy and it considers both the precision and the recall.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

Table 1 shows the performance evaluation results of the Naive Bayesian spam filter in terms of precision, recall and F1-score. Table 2 shows the performance using confusion matrix. In general, Naive

Table 1: Naive Bayes Model Precision

	Precision	Recall	F1-score	support
ham	0.96	1.00	0.98	961
spam	1.00	0.73	0.84	154
micro avg	0.96	0.96	0.96	1115
macro avg	0.98	0.86	0.91	1115
weighted avg	0.96	0.96	0.96	1115

Table 2: Confusion Matrix

	Predicted Spam	Predicted Ham
Actual Spam	970	0
Actual Ham	34	111

Table 3: Synonym list for the "Ringtone Club" message example.

Word	Synonyms
Ringtone	null
Club	null
Get	'acquire', 'become', 'go', 'let', 'have', 'receive', 'find', 'obtain', 'incur', 'arrive', 'come', 'bring', 'convey', 'fetch', 'experience', 'pay_back', 'pay_off', 'fix', 'make', 'induce', 'stimulate', 'cause', 'catch', 'capture', 'grow', 'develop', 'produce', 'contract', 'take', 'drive', 'aim', 'arrest', 'scram', 'buzz_off', 'bugger_off', 'draw', 'perplex', 'vex', 'stick', 'puzzle', 'mystify', 'baffle', 'beat', 'pose', 'bewilder', 'flummox', 'stupefy', 'nonplus', 'gravel', 'amaze', 'dumbfound', 'get_down', 'begin', 'start_out', 'start', 'set_about', 'set_out', 'commence', 'suffer', 'sustain', 'beget', 'engender', 'father', 'mother', 'sire', 'generate', 'bring_forth'
Singles	'single', 'bingle', 'one', '1', 'I', 'ace', 'unity'
Chart	'graph'
Mobile	'Mobile_River', 'nomadic', 'peregrine', 'roving', 'wandering', 'fluid'
Week	'hebdomad', 'workweek', 'calendar_week'
Choose	'take', 'select', 'pick_out', 'prefer', 'opt'
top	'top_side', 'upper_side', 'upside', 'peak', 'crown', 'crest', 'tip', 'summit', 'top_of_the_inning', 'acme', 'height', 'elevation', 'pinnacle', 'superlative', 'meridian', 'tip-top', 'whirligig', 'teetotum', 'spinning_top', 'cover', 'circus_tent', 'big_top', 'round_top', 'exceed', 'transcend', 'overstep', 'pass', 'go_past', 'clear', 'lead', 'top_out', 'pinch', 'top_off'
quality	'caliber', 'calibre', 'character', 'lineament', 'timbre', 'timber', 'tone', 'choice', 'prime', 'prize', 'select'
ringtone	null
message	'content', 'subject_matter', 'substance'
free	'free_people', 'liberate', 'release', 'unloose', 'unloosen', 'loose', 'rid', 'disembarrass', 'dislodge', 'exempt', 'relieve', 'discharge', 'disengage', 'absolve', 'justify', 'relinquish', 'resign', 'give_up', 'unblock', 'unfreeze', 'complimentary', 'costless', 'gratis', 'gratuitous', 'detached', 'spare', 'barren', 'destitute', 'devoid', 'innocent', 'liberal'
charge	null

Bayesian based classifier performs well in detecting spam messages without adversary scenarios.

4.4 Synonym Replacement

In the first adversary learning scenario, we use Natural Language Processing `nltk` module to find a list of synonyms for each word in any given message. Let us consider a message that is classified as *spam* by the spam filter:

"Ringtone Club: Get the UK singles chart on your mobile each week and choose any top quality ringtone! This message is free of charge."

WordNet interface of the `nltk` module returns a list of synonyms for the above message, as shown in Table 3.

Table 4: Construction of new messages from the original spam message (synonym replacements are in bold fonts) may fool machine learning based spam filters.

Modified Message	Cosine Similarity	Prediction
Ringtone Club: **acquire** the UK single **graph** on your **Mobile_River** each **hebdomad** and **take** any **top_side caliber** ringtone! This **content** is **free_people** of charge.	0.583	spam
Ringtone Club: **become** the UK **bingle graph** on your **nomadic** each **workweek** and **select** any **upper_side caliber** ringtone! This **subject_matter** is **liberate** of charge.	0.583	spam
Ringtone Club: **go** the UK **one graph** on your **peregrine** each **calendar_week** and **pick_out** any **upside character** ringtone! This **substance** is **release** of charge.	0.583	ham

New messages are then constructed from the original message using Algorithm 1. The results are shown in Table 4. The replacement of synonyms in the original message is highlighted in bold font. The cosine similarities are the same among all modified messages. However, when it comes to prediction, the first two messages are still classified as spam but the third message is classified as ham thus bypassing the filter.

4.5 Ham Words Injection

Let us consider a message that is classified as *spam* by the spam filter:

"Congratulations ur awarded 500 of CD vouchers or 125gift guaranteed & Free entry 2 100 wkly draw txt MUSIC to 87066 TnCs www.Ldew.com1win150ppmx3age16"

Some ham words such as "good", "great", "appreciate", etc., may be inserted in the beginning, middle or at the end of the message. The manipulated frequency of ham words may eventually alter the spam filter prediction. For example, the following adversarial example we found is classified as *ham* by the spam filter (injected words are highlighted in bold font):

*"Congratulations **good** ur awarded **good** 500 of CD vouchers or 125 **good** gift guaranteed **love** & Free entry 2 **good** 100 wkly draw txt MUSIC to 87066 TnCs www.Ldew.com1win150ppmx3age16 **good good good good good deal**"*

Let us consider another message that is classified as *spam* by the spam filter:

*"U 447801259231 have a secret admirer who is looking 2 make contact with U-find out who they R*reveal who thinks UR so special-call on 09058094597"*

It is interesting to observe that the spam filter may be confused if we replace the abbreviation form with their full form, as seen from the following adversarial example which is classified as *ham* by the spam filter (replacement is highlighted in bold font):

*"**you** 447801259231 have a secret admirer who is looking **to** make contact with **you** find out who they **are** reveal who thinks **you are** so special-call on 09058094597."*

In the above message we can observe that replacing the abbreviated words such as "U" as "you", "R" as "are", 'UR" as "your" causes the spam filter fail to detect the message as spam. The reason could be that the model learns that spam usually uses abbreviations. This may be an evidence that good writing style is actually awarding.

The similarity scores for the above two examples are 0.407 and 0.854. The first example has larger distance because of the repeated ham words in the message. The second example is quite similar since only abbreviated words are substituted.

4.6 Spam Word Spacing

To get a database of ham and spam words, we collected a list of 474 spam trigger words [12]. Let us first consider a message that is classified as *spam*:

"Text & meet someone sexy today. U can find a date or even flirt its up to U. Join 4 just 10p. REPLY with NAME & AGE eg Sam 25. 18 -msg recd@thirtyeight pence"

The following adversarial example is classified as *ham* (spacing of spam words is in bold font):

"Text & meet someone **s e x y** today. U can find a date or even **f l i r t** its up to U. Join 4 just 10p. REPLY with NAME & AGE eg Sam 25. 18 -msg recd@thirtyeight pence"

The cosine similarity is 0.861 suggesting that spacing does not change much the original message meaning. By adding spaces in the common spam words such as "sexy" and "flirt", it is possible to bypass the classifier. While a more sophisticated model may prevent this scenario, it would be difficult for the classifier since few models are capable of building more than bi-grams or tri-grams where the model combines the previous word with the present word and forms a corpus.

In summary, 60% of the times we are able to bypass the spam filter by using one of the three invasion techniques.

5 CONCLUSION

As we rely more and more on automated AI systems, the security of vast majority of machine learning based approaches is becoming increasingly important. In this paper, we study a machine learning based spam message filter and experiment three adversarial machine learning techniques to invade such spam filter. While in general the model performs very well in detecting spam and ham messages in adversary-free environment, such model can be bypassed easily when we put adversaries in the loop. While the results are promising, we understand the successfulness largely depends on the machine learning algorithms themselves, e.g., some adversarial examples that work on Naive Baysian may not work on other models. It is our future work to study the generality of adversarial learning for other security systems.

REFERENCES

[1] Ion Androutsopoulos, Georgios Paliouras, Vangelis Karkaletsis, Georgios Sakkis, Constantine D. Spyropoulos, and Panagiotis Stamatopoulos. 2000. Learning to Filter Spam E-Mail: A Comparison of a Naive Bayesian and a Memory-Based Approach. *Proceedings of the workshop "Machine Learning and Textual Information Access"* (Feb 2000), 1–12.

[2] Andrej Bratko, Gordon V. Cormack, Bogdan Filipic, Thomas R. Lynam, and Blaz Zupan. 2006. Spam Filtering Using Statistical Data Compression Models. *Journal of Machine Learning Research* 7 (Mar 2006), 2674–2698.

[3] Niddal H. Imam and Vassilios G. Vassilakis. 2019. A Survey of Attacks Against Twitter Spam Detectors in an Adversarial Environment. *Robotics* 8, 3 (2019).

[4] UCI Machine Learning. [n.d.]. SMS Spam Collection Dataset. *Kaggle: Your Machine Learning and Data Science Community https://www.kaggle.com/uciml/sms-spam-collection-dataset* ([n. d.]).

[5] Qin Luo, Bing Liu, Junhua Yan, and Zhongyue He. 2010. Research of a Spam Filtering Algorithm Based on Naive Bayes and AIS. In *International Conference on Computational and Information Sciences.* Chengdu, China, 152–155.

[6] Nuno Martins, Jose Magalhaes Cruz, Tiago Cruz, and Pedro Henriques Abreu. 2020. Adversarial Machine Learning Applied to Intrusion and Malware Scenarios: A Systematic Review. *IEEE Access* 8 (February 18 2020), 35403–35419.

[7] Vangelis Metsis, Ion Androutsopoulos, and Georgios Paliouras. 2006. Spam Filtering with Naive Bayes – Which Naive Bayes? *Third Conference on Email and Anti-Spam (CEAS), Mountain View, California USA* (July 2006).

[8] Blaine Nelson, Marco Barreno, Fuching Jack Chi, and Anthony D. Joseph. 2008. Exploiting Machine Learning to Subvert Your Spam Filter. *In Proceedings of First USENIX Workshop on Large Scale Exploits and Emergent Threats* (April 2008).

[9] Kunjali Pawar and Madhuri Patil. 2015. Pattern Classification under Attack on Spam Filtering. In *IEEE International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN).* Kolkata, India, 197–201.

[10] JUNYAN PENG and PATRICK P. K. CHAN. 2013. REVISED NAIVE BAYES CLASSIFIER FOR COMBATING THE FOCUS ATTACK IN SPAM FILTERING. *Proceedings of the IEEE International Conference on Machine Learning and Cybernetics* (july 2013), 610–614.

[11] Rohit Kumar Solanki, Karun Verma, and Ravinder Kumar. 2015. Spam filtering using hybrid local-global Naive Bayes classifier. In *International Conference on Advances in Computing, Communications and Informatics (ICACCI).* Kochi, India, 829–833.

[12] Pete Thompson. 2020. The Ultimate SPAM Trigger Words List: 475 Keywords to Avoid in 2020. *Automational* (2020). https://www.automational.com/spam-trigger-words-to-avoid/

[13] Fergus Toolan and Joe Carthy. 2010. Feature Selection for Spam and Phishing Detection. *IEEE eCrime Researchers Summit* (October 2010), 1–12.

[14] Steve Webb, Subramanyam Chitti, and Calton Pu. 2005. An Experimental Evaluation of Spam Filter Performance and Robustness Against Attack. In *IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing.* San Jose, CA.

[15] Qijia Wei. 2018. Understanding of the Naive Bayes Classifier in Spam Filtering. In *6th International Conference on Computer-Aided Design, Manufacturing, Modeling and Simulation (CDMMS)*, Vol. 1967.

[16] Tianda Yang, Kai Qian, Dan Chia-Tien La, Kamal AI Nasr, and Ying Qian. 2015. Spam Filtering using Association Rules and Naive Bayes Classifier. *IEEE International Conference on Progress in Informatics and Computing (PIC)* (Dec 2015), 638–642.