A Machine Learning approach towards Phishing Email Detection

CEN-Security@IWSPA 2018

Harikrishnan NB, Vinayakumar R, Soman KP
Center for Computational Engineering and Networking(CEN),
Amrita School of Engineering, Coimbatore
Amrita Vishwa Vidyapeetham, India
harikrishnannb07@gmail.com

Abstract

Email is a platform where we communicate, exchange ideas between each other. In today's world email plays a key role irrespective of the field. In such a scenario, phishing mails are one of the major threats in today's world. These e-mails "seems" like legitimate but leads the users to malicious sites. As a result the user or organization or institution end up as the prey of the online predators. In order to tackle such problems, several statistical methods have been applied. In this paper we make use of distributional representation namely TF-IDF for numeric representation of phishing mails. Also a comparative study of classical machine learning techniques like Random Forest, AdaBoost, Naive Bayes, Decision Tree, SVM.

1 Introduction

In today's world communication plays a key role in all aspects of life. Email is a common platform used by people for faster and efficient communication. Email has become an inevitable part of everyday life. Due to the advancement in this era of digitization the dependency on email has been increasing day by day. The increasing dependency calls for a way to manage the huge amount of data or emails. The emails conveyed include important as well as phishing emails. Phishing emails often leads to malicious websites and results in sharing personal details to the attackers. In order to thwart these situations spam and phishing email classifiers are widely used. Blacklisting which comes under the category of list based filters is a popular method to thwart phishing emails. It achieves this by blocking emails from a list of sender's that are in the blacklist. Blacklist consists of records of IP address and email address of malicious users. When a new emails arrives, the spam and phishing email filter checks the IP and email address with that provided in the blacklist and decides whether the email has to be marked as phishing or not. Other list based filters include whitelist-which allows emails from senders that is provided by the user. Other popular methods include filters based on contents. This includes word based filters, heuristic filters, Bayesian filters. Word based filters blocks emails with certain specific words. The main drawback of this method is its failure to classify new malicious email. In order to update the list human intervention is required

Phishing email is a common name that represents spam emails that has malicious intentions. Phishing emails are a potential danger especially to multinational companies, banking sector and even hospitals. Phishing emails are also used by hackers to inject malware into the system. The recent ransomware attack [KRB+15] is the best example for this. These phishing emails seems like legitimate but contains malicious contents which can steal ones valuable details like account number, credit/debit card details etc. In such a situation a model has to be developed which can detect and classify phishing emails very efficiently. The traditional methods relies on human intervention. This calls for an automation in recognizing emails as either phishing or not. In such situations research moves in

In: R. Verma, A. Das (eds.): Proceedings of the 1st AntiPhishing Shared Pilot at 4th ACM International Workshop on Security and Privacy Analytics (IWSPA 2018), Tempe, Arizona, USA, 21-03-2018, published at http://ceur-ws.org

the direction of machine learning and deep learning.

Recent developments in the field of machine learning and deep learning, have shown promising results in the field of Computer Vision, Natural Language Processing, Cyber security. etc. Taking this into account we use a machine learning based model like Decision tree, Logistic Regression, Random forest, Naive Bayes, KNN, AdaBoost, SVM in classifying email as either phishing or legitimate. The proposed method uses SVD (singular value decomposition), NMF (Non-negative Matrix Factorization) for feature extraction and dimensionality reduction. We have used TFIDF (Term Frequency Inverse Document Frequency) for numeric representation of words.

The paper is structured as follows: Section 2 represents related works, Section 3 discuses dataset description, Section 4 highlights the methodology used, Section 5, 6, 7 represents results, conclusion and acknowledgement respectively.

2 Related Work

Phishing email detection can be treated as a sub problem of spam detection. For several years spam detection has been a rich area of research. [AKCS00], [Sch03], [CL06] are examples of earlier works on anti-spam filters. The work done specifically on phishing email detection is comparatively less compared to spam detection. The dataset commonly used for most of the research related to Phishing email is PhishingCorpus [Naz10], [SVKS15], [BVP]. PhishingCorpus consist of a group of hand-screened emails [GNN11] which makes the dataset challenging. The existing learning based approaches are presented in a structured overview in [BB08]. Currently, various experts are tacking the problem of phishing email classification in the perspective of text classification [BB08]. In [CNU06] performed phishing email detection by identifying structural features from the emails. These features are passed to SVM for detecting phishing emails. In [BCP$^+$08] has proposed two methods, adaptive Dynamic Markov Chains (DMC) and latent class-topic model to classify emails. The adaptive Dynamic Markov Chains gave similar performance when compared to standard version while using two thirds less of the memory. In [ANNWN07] has proposed machine learning based models like logistic regression, SVM, random forest for classifying emails as either spam or legitimate. Also [AGA$^+$13] has mentioned the types of phishing attacks and classification. However they have not incorporated the exploration of available datasets and feature engineering techniques. Researchers has also analyzed the classfication of emails based on the contents. This paper uses TF-IDF representation followed by dimensionality reduction for capturing major contributing factors in the dataset and also for reducing the computational cost. This is then passed to classical machine learning techniques for classifying the data as either legitimate or normal. Researchers has also moved in the direction of applying deep learning techniques to classify URL's as benign and malicious URL's [VSP18b], [VSP18a]. In [VSPSK18], [VSP17] authors have used deep learning techniques to classify and evaluate domain generation algorithm.

3 Dataset description

The shared task consists of two tasks. Task 1 is Email with headers and Task 2 is Email with no headers. The dataset details [EDMB$^+$18] [EDB$^+$18] is provided in the table below:

Table 1: Training Dataset details

Training Dataset	Legitimate	Spam	Total
With header	4082	501	4583
With No header	5088	612	5700

Table 2: Testing Dataset details

Testing Dataset	Data Samples
With header	4195
With No header	4300

4 Methodology

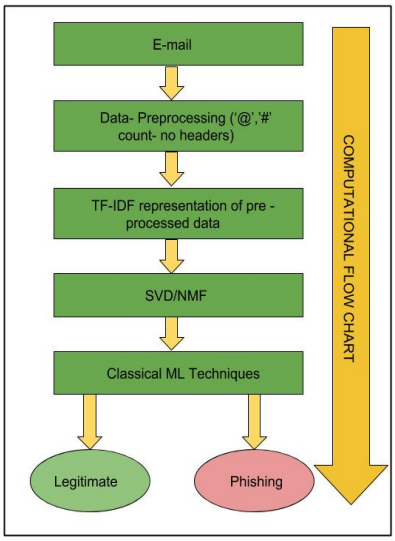

Figure 1: Proposed Architecture

4.1 Dataset representation

The proposed architecture is provided in Figure 1. The goal of the task is as follows:

- Given a set of emails represented as $D = [e_1, e_2, ...e_n]$ and its labels like $C = [c_1, c_2, ...c_n]$. The labels are either 0 or 1. The machine learning model used will learn the patterns that maps the train data into its corresponding labels. After the learning, the model is used to predict the labels for test data.

In order to represent data in numeric format we used TF-IDF representation. TF-IDF (Term Frequency Inverse Document Frequency) is used for both the tasks. TF-IDF represents the importance of a word in a corpus. The TF-IDF representation is followed by SVD/ NMF for feature selection and dimensionality reduction. We have used train-test split and chose 33% of training data as validation data for evaluating the performance of the model

We have evaluated the performance of TF-IDF representation and TF-IDF + SVD/NMF representation for the validation data. For TFIDF + SVD/NMF, the rank is taken as 30 i.e, the number of columns of the train and test data matrix will be 30 due to dimensionality reduction. The performance of TF-IDF + SVD/NMF with no of columns as 30 after dimensionality reduction was similar to the performance of TFIDF representation of validation data. This numeric representation for the data is passed to different machine learning algorithms.

4.1.1 Data representation for with headers:

- TF-IDF representation of data. The vocabulary is build using train and test data.
- SVD/NMF for feature extraction and dimensionality reduction
- Step 2 is followed by applying classical ML techniques like Decision Tree, Random Forest, AdaBoost, KNN, SVM

4.1.2 Data representation for with no headers:

- Data Preprocessing- Data preprocessing involves counting the number of '@', '#' symbol in each data sample. Then '@' and '#' counts are removed from orginal corpus
- TF-IDF representation of data, followed by appending the '@' count and '#' count.
- SVD/NMF for feature extraction and dimensionality reduction
- Step 3 is followed by applying classical ML techniques like Decision Tree, Random Forest, AdaBoost, KNN, SVM

In this paper we have used classical machine learning techniques like Decision Tree, K- Nearest Neighbors, Logistic Regression, Naive Bayes, Random Forest, SVM. The metrics for understanding the performance are the following:

1. Accuracy
2. Precision
3. Recall
4. F1-Score

The techniques used for feature extraction and dimensionality reduction are NMF and SVD. In [LS99] describes the details of Non Negative Matrix Factorization well. TFIDF matrix is passed as input to NMF and a group of topics is generated. These represents a weighted set of co-occurring terms. The topics identified acts as a basis by providing an efficient way of representation to the original corpus. NMF is found useful when the data attributes are more and is used as a feature extraction technique.

SVD aka singular value decomposition, decomposes the TFIDF matrix ($\mathbf{T}$) into 3 matrices. They are U, Σ, V^T, U represents the orthonormal eigenvectors of AA^T , represents a diagonal matrix and its diagonal entries are the singular values, V^T represents the orthogonal eigenvectors of A^TA. SVD is a powerful tool and has many application in the field of signal processing and image processing. SVD is mainly used for dimensionality reduction and for representing important features. The product of $U\Sigma$ is used for extracting the features. In all the cases the rank is assumed as 30. So the size of train and test matrix will shrink to (no of data samples x 30). These extracted features are passed to different classical machine learning techniques

5 Results

This section provides details of the accuracy, precision, recall, F1-score with respect to training data. The following tables describes the performance of each classical machine learning techniques for the formulated binary classification problem to detect whether an email is phishing or legitimate. We have used train-test split (scikit learn) to split the training data into training and validation. We have used 33% of training data for validation. Table 3, 4, represents metrics for validation for sub-task 1 (no header) and sub-task 2 (with

Table 3: Results for validation data for sub-task 1

TFIDF	Accuracy (%)	Precision	Recall	F1-Score
Decision Tree	96.5	0.832	0.837	0.835
KNN	97.6	0.921	0.837	0.877
Logistic Regression	96.8	0.986	0.704	0.821
Naive Bayes	94.7	0.77	0.694	0.733
Random Forest	97.1	1.0	0.719	0.837
AdaBoost	97.7	0.927	0.842	0.882
SVM	98.7	0.978	0.898	0.936

Table 4: Results for validation data for sub-task 2

TFIDF	Accuracy (%)	Precision	Recall	F1-Score
Decision Tree	99.9	0.994	1.00	0.997
KNN	99.6	0.982	0.982	0.982
Logistic Regression	98.9	1.0	0.901	0.948
Naive Bayes	98.4	1.00	0.860	0.925
Random Forest	99.9	1.0	0.994	0.997
AdaBoost	99.9	1.0	0.994	0.997
SVM	99.9	1.0	0.988	0.994

Table 5: Results for validation data for TFIDF+SVD representation

TFIDF+SVD	Task	Accuracy (%)	Precision	Recall	F1-Score
Decision Tree	sub-task 1	99.6	0.982	0.982	0.982
KNN	sub-task 1	99.9	1.0	0.988	0.994
Logistic Regression	sub-task 1	98.9	1.0	0.901	0.948
Naive Bayes	sub-task 1	99.3	0.949	0.988	0.968
Random Forest	sub-task 1	99.8	1.0	0.982	0.991
AdaBoost	sub-task 1	99.9	1.0	0.988	0.994
SVM	sub-task 1	99.9	1.0	0.988	0.994
Decision Tree	sub-task 2	96.1	0.809	0.821	0.815
KNN	sub-task 2	97.8	0.943	0.837	0.886
Logistic Regression	sub-task 2	96.2	0.977	0.653	0.783
Naive Bayes	sub-task 2	66.7	0.236	0.980	0.380
Random Forest	sub-task 2	98	0.982	0.827	0.898
AdaBoost	sub-task 2	97.6	0.912	0.847	0.878
SVM	sub-task 2	97.6	0.917	0.842	0.878

Table 6: Results for validation data using TFIDF+NMF representation

TFIDF+NMF	Task	Accuracy (%)	Precision	Recall	F1-Score
Decision Tree	sub-task 1	99.7	0.988	0.982	0.985
KNN	sub-task 1	99.7	0.994	0.982	0.988
Logistic Regression	sub-task 1	88.7	0.0	0.0	0.0
Naive Bayes	sub-task 1	98.0	0.851	1.0	0.919
Random Forest	sub-task 1	99.9	1.0	0.994	0.997
AdaBoost	sub-task 1	99.9	1.0	0.988	0.994
SVM	sub-task 1	99.9	1.0	0.994	0.997
Decision Tree	sub-task 2	97.0	0.868	0.837	0.852
KNN	sub-task 2	97.8	0.929	0.852	0.888
Logistic Regression	sub-task 2	89.6	0.0	0.0	0.0
Naive Bayes	sub-task 2	61.3	0.210	0.985	0.346
Random Forest	sub-task 2	97.7	0.932	0.837	0.882
AdaBoost	sub-task 2	97.2	0.914	0.811	0.859
SVM	sub-task 2	97.0	0.912	0.791	0.841

Table 7: Results for test set using TFIDF+SVD representation for sub-task 1 and sub-task 2

TFIDF+SVD	Task	Accuracy (%)	Precision	Recall	F1-Score
Decision Tree	sub-task 1	76.232	0.877	0.851	0.864
KNN	sub-task 1	83.953	0.883	0.943	0.912
Logistic Regression	sub-task 1	81.069	0.880	0.911	0.895
Naive Bayes	sub-task 1	82.04	0.885	0.916	0.901
Random Forest	sub-task 1	87.97	0.88	0.988	0.936
AdaBoost	sub-task 1	83	0.883	0.932	0.907
SVM	sub-task 1	45.46	0.864	0.458	0.59
Decision Tree	sub-task 2	76.92	0.903	0.826	0.863
KNN	sub-task 2	82.145	0.882	0.920	0.900
Logistic Regression	sub-task 2	87.50	0.880	0.992	0.933
Naive Bayes	sub-task 2	88.15	0.881	0.999	0.937
Random Forest	sub-task 2	87.55	0.886	0.984	0.933
AdaBoost	sub-task 2	85.125	0.881	0.959	0.919
SVM	sub-task 2	68.93	0.863	0.769	0.813

Table 8: Results for test set using TFIDF+NMF representation for sub-task 1 and sub-task 2

TFIDF+NMF	Task	Accuracy (%)	Precision	Recall	F1-Score
Decision Tree	sub-task 1	84	0.893	0.931	0.911
KNN	sub-task 1	87.255	0.892	0.974	0.9315
Logistic Regression	sub-task 1	88.95	0.88	1	0.941
Naive Bayes	sub-task 1	68.97	0.901	0.730	0.807
Random Forest	sub-task 1	86.90	0.887	0.976	0.929
AdaBoost	sub-task 1	86.06	0.88	0.964	0.924
SVM	sub-task 1	88	0.888	0.989	0.936
Decision Tree	sub-task 2	81.12	0.964	0.815	0.883
KNN	sub-task 2	90.29	0.925	0.967	0.946
Logistic Regression	sub-task 2	88.17	0.881	1	0.937
Naive Bayes	sub-task 2	84.00	0.916	0.901	0.908
Random Forest	sub-task 2	80.619	0.945	0.827	0.882
AdaBoost	sub-task 2	77.044	0.932	0.797	0.858
SVM	sub-task 2	89.964	0.920	0.970	0.944

header). The results in Table 3 and 4 corresponds to the TFIDF representation of the data. Similarly Table 5 and 6 represents the evaluation metrics for validation data for sub-task 1 (no header) and sub-task 2 (with header) with TFIDF + SVD/NMF representation respectively. When calculated the training accuracy Decision Tree and Random Forest outperformed almost in all cases. The performance of TFIDF and TFIDF +SVD/NMF representation is almost similar from the results obtained in Table 3, 4, 5, 6. This motivates us to go for dimensionality reduction. Since the number of singular values used are 30, the pre-processed data set size will be (no of rows, 30) Table 7, 8 represents metrics for test set. Table 7 represents the metrics for TFIDF + SVD representation for sub-task 1 and 2 test set. Similarly Table 8 represents the metrics for TFIDF + NMF representation for sub-task 1 and 2 test set.

6 Conclusion

In this paper we used TFIDF+ SVD and TFIDF + NMF representations followed by ML techniques for classifying emails as either legitimate or phishing. The performance of Decision Tree and Random Forest was the highest in the case of training accuracy. But the test data results for decision tree and random forest mentions the case of overfitting. The overfitting is because the dataset is highly unbalanced. Also both the sub-tasks belong to the unconstrained category (which means we can use any other data sets during training). The given datasets for both the sub-tasks are highly imbalanced. Even though the tasks are unconstrained, we haven't used any other external sources. With highly, imbalanced data sets, we are able to achieve considerable phishing email detection rate in both the sub-tasks. The phishing email detection rate of the proposed methodology can be easily enhanced by adding additional extra data sources. This will be considered as one of the significant direction towards the future work. Also due to computational constraints, the authors couldn't try for deep learning based methods. This can also be taken up as a future work.

Acknowledgement

This research was supported in part by Paramount Computer Systems. We are also grateful to NVIDIA India, for the GPU hardware support to the research grant. We are grateful to Computational Engineering and Networking (CEN) department for encouraging the research.

References

[AGA+13] Ammar Almomani, BB Gupta, Samer Atawneh, A Meulenberg, and Eman Almomani. A survey of phishing email filtering techniques. *IEEE communications surveys & tutorials*, 15(4):2070–2090, 2013.

[AKCS00] Ion Androutsopoulos, John Koutsias, Konstantinos V Chandrinos, and Constantine D Spyropoulos. An experimental comparison of naive bayesian

and keyword-based anti-spam filtering with personal e-mail messages. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 160–167. ACM, 2000.

[ANNWN07] Saeed Abu-Nimeh, Dario Nappa, Xinlei Wang, and Suku Nair. A comparison of machine learning techniques for phishing detection. In *Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit*, pages 60–69. ACM, 2007.

[BB08] Enrico Blanzieri and Anton Bryl. A survey of learning-based techniques of email spam filtering. *Artificial Intelligence Review*, 29(1):63–92, 2008.

[BCP+08] Andre Bergholz, Jeong Ho Chang, Gerhard Paass, Frank Reichartz, and Siehyun Strobel. Improved phishing detection using model-based features. In *CEAS*, 2008.

[BVP] Barathi Ganesh Hullathy Balakrishnan, Anand Kumar Madasamy Vinayakumar, and Soman Kotti Padannayil. Nlp cen amrita@ smm4h: Health care text classification through class embeddings.

[CL06] Victor Cheng and Chun Hung Li. Personalized spam filtering with semi-supervised classifier ensemble. In *Proceedings of the 2006 IEEE/WIC/ACM international Conference on Web intelligence*, pages 195–201. IEEE Computer Society, 2006.

[CNU06] Madhusudhanan Chandrasekaran, Krishnan Narayanan, and Shambhu Upadhyaya. Phishing email detection based on structural properties. In *NYS Cyber Security Conference*, volume 3, 2006.

[EDB+18] Ayman Elaassal, Avisha Das, Shahryar Baki, Luis De Moraes, and Rakesh Verma. Iwspa-ap: Anti-phising shared task at acm international workshop on security and privacy analytics. In *Proceedings of the 1st IWSPA Anti-Phishing Shared Task*. CEUR, 2018.

[EDMB+18] Ayman Elaassal, Luis De Moraes, Shahryar Baki, Rakesh Verma, and Avisha Das. Iwspa-ap shared task email dataset, 2018.

[GNN11] Hugo Gonzalez, Kara Nance, and Jose Nazario. Phishing by form: The abuse of form sites. In *Malicious and Unwanted Software (MALWARE), 2011 6th International Conference on*, pages 95–101. IEEE, 2011.

[KRB+15] Amin Kharraz, William Robertson, Davide Balzarotti, Leyla Bilge, and Engin Kirda. Cutting the gordian knot: A look under the hood of ransomware attacks. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 3–24. Springer, 2015.

[LS99] Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788, 1999.

[Naz10] J Nazario. Phishingcorpus homepage. *ed: Retrieved February*, 2010.

[Sch03] Karl-Michael Schneider. A comparison of event models for naive bayes anti-spam e-mail filtering. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pages 307–314. Association for Computational Linguistics, 2003.

[SVKS15] Shriya Se, R Vinayakumar, M Anand Kumar, and KP Soman. Amrita-cen@ sail2015: sentiment analysis in indian languages. In *International Conference on Mining Intelligence and Knowledge Exploration*, pages 703–710. Springer, 2015.

[VSP17] R Vinayakumar, KP Soman, and Prabaharan Poornachandran. Deep encrypted text categorization. In *Advances in Computing, Communications and Informatics (ICACCI), 2017 International Conference on*, pages 364–370. IEEE, 2017.

[VSP18a] R Vinayakumar, KP Soman, and Prabaharan Poornachandran. Detecting malicious domain names using deep learning approaches at scale. *Journal of Intelligent & Fuzzy Systems*, 34(3):1355–1367, 2018.

[VSP18b] R Vinayakumar, KP Soman, and Prabaharan Poornachandran. Evaluating deep

learning approaches to characterize and classify malicious urls. *Journal of Intelligent & Fuzzy Systems*, 34(3):1333–1343, 2018.

[VSPSK18] R Vinayakumar, KP Soman, Prabaharan Poornachandran, and S Sachin Kumar. Evaluating deep learning approaches to characterize and classify the dgas at scale. *Journal of Intelligent & Fuzzy Systems*, 34(3):1265–1276, 2018.