A.R.E.S : Automatic Rogue Email Spotter

Crypt Coyotes

Vysakh S Mohan, Naveen J R, Vinayakumar R, Soman KP
Center for Computational Engineering and Networking(CEN),
Amrita School of Engineering, Coimbatore
Amrita Vishwa Vidyapeetham, India
vsmo92@gmail.com,naveenaksharam@gmail.com

Abstract

Be it formal or casual, email is undoubtedly the most popular means of communication in modern times. Their popularity owes to the fact that they are reliable, fast and more over free to use. One issue that plagues this otherwise solid technology is phishing emails received by users. Phishing emails have always bothered users as it's a huge waste of storage, time, money and resource to any user. Many previous attempts to eradicate or at least block phishing emails have been deemed futile. This work uses word embedding as text representation for supervised classification approach to identify phishing emails. Ruled based and machine learning models with feature engineering were attempted but failed due to the ever increasing ways of threats and lack of scalability of the model. Deep learning based models have shown to surpass the older techniques in spam email detection. This work aims at attempting the same using a CNN/RNN/MLP network with Word2vec embeddings on phishing email corpus, where Word2vec helps to capture the synaptic and semantic similarity of phishing and legitimate emails in an email corpus. This work aims to show the abilities of word embedding have to solve issues related to cybersecurity use cases.

1 Introduction

Internet and staying connected through it is what distinguishes this era from the previous. More and more people rely on the internet for their communication as well as data transaction requirements. Email has revolutionized the way people communicate over the web. From its inception, electronic mails have outgrown its real world counterpart to become mainstream and serve as both casual and official way of passing a message. Now we have several service providers offering email platforms for free and with a plethora of features. This means that the number of people taking advantage of these services have grown dramatically. This mass adoption is one aspect any malignant adversary could use to his benefit. Such malignant emails are called spam[CM01], and they are unsolicited as well as junk info usually unwanted for the user. They are commonly characterized by the following: they are mass mailed, may contain explicit content, useless advertisements, fraudulent, may contain hidden links to phishing websites etc. On a personal front the user could face issues like, annoyance due to irrelevant info, unwanted use of bandwidth, waste of storage, makes the communication channel less productive via loss of time sorting junk mails, unnecessary use of computing power, causes spread of viruses, loss of money via phishing etc.

These issues have brought immense focus on safety of users against spam emails. Massive pool of users using these platforms is one reason for it being targeted more often. It is an inexpensive means to gain access to millions of people, which forces adversaries to target it more often. The most dangerous type of emails are the spam emails[KRA^{+}07]. It may be via a spam email server or from personal servers containing malicious URLs that could direct the users to phishing sites. This is a challenging task and many solutions have been devised to solve this problem over the

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

past few years, but they all come with some downsides. One reason it gets challenging is the variety of ways in which the attacker can serve a spam email. A frequently used method is the blended attack. Malware delivery through such attacks may vary. Usually the email itself may not contain the malware, but possibly contain a link to some compromised website. These emails may look normal, but would contain a mix of legitimate as well as malicious content. A former research by IBM's X-Force team, found that more than 50% of the emails produced worldwide are fraudulent. These figures are going to increase in the subsequent years.

One reason such attacks are successful is the carelessness from the generic user. Most internet users are illiterate when it comes to cybersecurity and they simply ignore the safety precautions that need to be exercised in the online space. There are no sure shot ways to check if a person has been a victim to such attacks, but can be prevented by being a bit cautious. You could check the email headers and check for grammatical mistakes. But these may not be sufficient when the scale of such attacks escalates. These type of states require some automated solution to detect spam email.

Emails headers can help to a certain extent. They can be used as features to some machine learning based classifiers[LT04, S+09]. The advantage of using header features compared to body features have been detailed in[ZZY04]. Header features like sender address, message ID etc. were used in[WC07] to make the detection.

Most of the popular machine learning techniques consists of two steps: obtain the proper features representation from the data and use these features for learning and predicting the system. First step focuses on extracting useful info from the given URL, which is stored as a vector so that the algorithm can fit different machine learning based models in it. Different categories of features have been taken[SLH17]. Lexical features, content features, host based features and context features are some of the popular ones. An algorithm requires some form of mathematical representation to work with. This work uses Word2vec embedding methods for effective representation of the data.

Spam filtering is a supervised classification problem where the problem is considered as a binary classification task with 2 classes: legitimate (good) emails and spam emails. Tretyakov used methods like naive bayes and K-NN machine learning algorithms for spam detection[Tre04], which doesn't deal with feature selection but beneficial for beginners. Spam detection or automatic email filtering starts with statistical approaches primarily. The development began with popular naive bayes approaches, which reduced the problem into a space where dependencies between the data and co-relation issues are ignored[SDHH98], that is, the multi variate nature of the problem breaks down to a uni-variate one without compromising on accuracy. Different authors have tried to incorporate modifications on top of the naive bayes pipeline, but the approach was unable to find the correlation between words and the algorithm failed in certain tasks. In 2004 Chih-Chin Lai and Tsai[LT04] introduced the TF-IDF, K-NN and SVM to overcome the issues in the email filtering task. SVM, TF-IDF got a satisfactory result while K-NN got worst result among them. Blanzieri and Bryl came up with feature extraction methods in[BB08], along with SVM. During this time, unsupervised machine leaning techniques were also developed. Data were clustered into spam and ham. Whissell and Clarke[WC11] in 2011, came up with a novel research on spam clustering, which attained state of art result compared to all the previous methods. Since the spam filtering is a diverse area, ensemble methods (combining different algorithms on same problem), like boosting and bagging[GGWM+10], are applied to get effective classification. Caruana and Li, (2012) focused[CL12] on distributed computuing paradigram using SVM and ANN by removing the interoperability and implementation issues.

Machine learning models usually rely on some sort of engineered features that are generated from the data and has been proved to surpass the accuracy of its predecessors in spam email classification[FRID+07, AAY11], whereas, very few machine learning models for phishing emails exist today and most of them are in their infancy. With acquired domain knowledge, various feature engineering strategies are employed on the data to build the model[SAZ18], [PHS18], [VH13], [VSH12], [MG18], [HDC+18], [MBA18]. A main plus to this method is the reduced effort to train the classifier rather than developing complex rules for a filter. This feature engineering method could also deem the system vulnerable to manipulation and the model may not scale well to newer threats. Deep learning models can be used to overcome this issue as they learn the features themselves and modify it according to newer inputs. On top of that these models are comparatively more accurate and scalable. Nowadays deep learning models combined with word embeddings have given good performance for various cybersecurity usecases[VSP18a], [VSP18b], [VSP17], [LF17], [SKP18]. This motivated the use of word embeddings with deep learning models like Multi-Layer Perceptron (MLP), Recurrent Neural Network (RNN), Convolutional Neural Networks and Long Short Term Memory (LSTM).

2 Background

This section details the theory behind various deep learning models used.

2.1 Word2vec

Word2vec is a model proposed by Mikalov[MSC$^+$13] to learn the word embedding which is inspired by distributed representation introduced by Hinton[HMR$^+$86], but in the Word2vec framework, word representation is learned using a shallow neural network. The fundamental assumption in word embedding or distributional methods is that, words with similar sense tends to happen in similar context and they capture the similarity between words[BG17], [BG18]. Word2vec is a popular model to generate word embeddings on text data. They have the ability to reproduce linguistic context of words through training their shallow two layer architectures. The input to the Word2vec model may be a huge corpus and the generated outputs are vectors in some multi-dimensional space, with each unique word in the corpus have a corresponding vector associated with it. This makes learning the word representation significantly faster than the previous methods. In the Word2vec framework the distributed representation of the words in the vocabulary is learned in an unsupervised way. Learning can be done via two architectures like skip-gram and continuous bag of words.

$$\frac{1}{N}\sum_{n=1}^{N}\sum_{-sks,j\neq 0} logp(Q_{n+k}|Q_k) \tag{1}$$

Skip-gram method tries to maximize the average probability value of the word sequence Q_1,Q_2,...Q_N. Here 's' indicates training context size that is directly related to the center word Q_n and $p(Q_{n+k}|w_k)$ is softmax function. In the skip-gram model, the context or surrounding word is predicted given the centre word as the input and in Continuous Bag of Words(CBOW) model, given the surrounding words the centre word is predicted.

2.2 Convolutional neural Nets (CNN)

CNN is commonly used for computer vision tasks, where their local receptive field is advantageous for feature learning in images. CNN models are also used for text classification tasks. CNN can be thought of as an artificial neural network that has the ability to pick out or detect patterns and make sense out of them. These pattern detection makes CNN useful for data analysis. CNN has hidden layers called convolutional layers are a tad bit different from MLP. For each convolutional layers, the number of filters needs to be specified, which then slides over the entire rows and columns of the matrix. In this matrix each individual row is a vector representing one word, more accurately speaking, these are word embedding models like Glove[1] or Word2vec[2]. This work used Word2vec model before applying CNN in this task. CNN performs well on sequential data with faster training times and is exceptional for predictive analysis. CNN normally consist of an input layer followed by convolutional layers, maxpooling layers for dimensionality reduction purpose and fully connected layers with a specific nonlinear activation function (*ReLU* in this work). In this phishing email detection task (text based), one dimensional maxpooling layers and fully connected layers are used. Filters used in this network model slides above the embedding vector to output a continuous value at each step. This outputs better representations of the word vectors. For text based applications 1D CNN is used.

2.3 Multi-layer perceptron (MLP)

Rosenblatt introduced the concept of a single perceptron. Multi-layer perceptron (MLP) is typically a network of perceptrons or simple neurons. MLP consists of one input and output layer. Dimensions of input output nodes depends on the no of sample vectors and the no of label vectors present in the input data. In between these two layers, many hidden layers are present. There exist layers where the output is being fed as input to the following hidden layers and each unit does a relatively straight forward computation. It takes input X multiplies it by a weight W, performs a summation and passes all of that through an activation function to yield the output. Perceptrons compute a score or a single output from sequential inputs that are usually real valued. This calculated score is used for backward pass, where cost function is calculated by matching wrongly predicted output to the truth label value, and is expressed as root mean square (RMS) error value. This RMS error is minimized using gradient descent technique and optimum weight and base value is figured out from this network model. It uses activation functions like *sigmoid* or *tanh* to produce the output. One nature of MLP is the fully connected architecture within its deep layers.

2.4 Recurrent neural network (RNN)

The problem associated with MLP and CNN model is that every input and outputs vectors are independent. Or in other words above models can't capture the sequential info between the words. In phishing email

[1] https://nlp.stanford.edu/projects/glove

[2] https://www.tensorflow.org/tutorials/Word2vec

Table 1: Hyper Parameter for Word2vec Model

Hyperparameter		
Batch-Size	250	The number of training samples required
Embedding-Size	300	Word vector dimension
Skip-Window	7	Context window, five words before and after each word
Num-skips	12	How many prediction pairs are selected from the window
Num-sampled	128	Number of negative samples
Learning rate	0.1	Determines how quickly or slowly model update the parameter
n-epoch	50	No of (forward+backward pass)

detection task it is highly useful to identify the associated words for classification purposes. RNN model is popular in time series and sequence data analysis. It can take variable size inputs and return a variable size output. State of recurrent NN at time 'T' is a function of its old state and the input at the time 'T'. Since it is storing previous state of system we can say that RNN has a 'memory' to capture sequential info between words. Recurrent neural net is a varied iteration of feed forward nets. The cyclic connections between the neurons makes way for results from previous time step to compute the current state, in a way remembering the temporal information about the input data. This makes RNN learn well on data with long term dependency, like for natural language processing and speech processing applications.

3 DATASET DESCRIPTION

The dataset[EDMB^{+}18] used is provided at the 4^{th} ACM International Workshop on Security and Privacy Analytics shared task[EDB^{+}18]. The task was to detect phishing emails. Details of the dataset is shown in Table2 & 3

Table 2: Training Dataset details

Category	Legitimate	Phishing	Total
With No header	5088	612	5700
With header	4082	501	4583

Table 3: Testing Dataset details

Training Dataset	Data Samples
With No header	4300
With header	4195

4 Experiments and Result

The proposed tool is christened A.R.E.S which stands for Automatic Rogue Email Spotter. A detailed visualization of the model is shown in Fig 1. The architecture is a combination of word embedding with a CNN, RNN, and MLP. This task is categorized into 2 subs tasks, which are emails with 'no header' and 'with header'. We didn't extract any other features from the header and the methodology used for conversion of raw email samples to feature vectors the same for both the sub tasks. In both the sub tasks, the raw email corpus is fed to the embedding layer that uses Word2vec model to generate distributed word embedding. The learned word embedding model is used to represent the input data, which is then fed to a deep learning models. The hyperparameters used to create Word2vec model is detailed in Table 1.

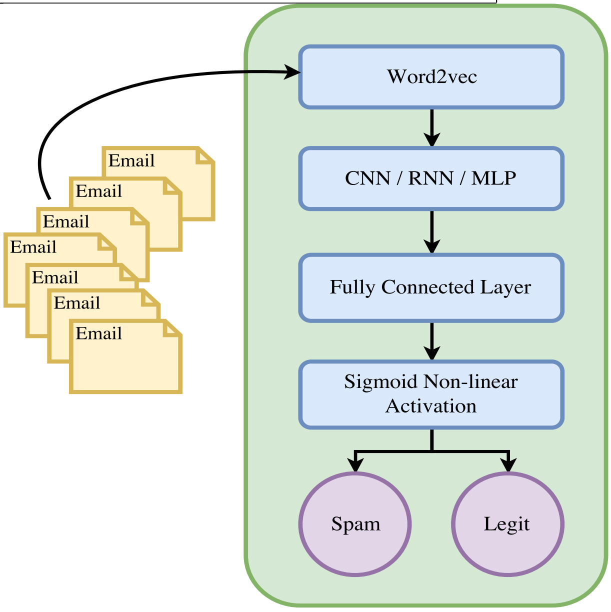

Figure 1: Proposed Architecture

The deep learning models learn additional features which will be pushed to the fully connected layer. Previous work on similar problem suggests to use RNN to solve such tasks, but in order to have a better analysis on the performance of different models we incorporated CNN and MLP to this work. Finally, due to the binary nature of this task we used *sigmoid* to classify legitimate emails from the phishing based on its threshold and used binary cross entropy for loss reduction.

From the statistics shown in Table 4 and 5, the word embedding model along with an MLP network gives a commendable score for both the sub tasks. Further, when the same word embedding model is passed

Table 4: Statistics of training results

Method	**Task**	**10-fold cross validation accuracy**
Word embedding + MLP	Sub task 1	0.921
Word embedding + CNN	Sub task 1	0.952
Word embedding + RNN	Sub task 1	0.951
Word embedding + MLP	Sub task 2	0.901
Word embedding + CNN	Sub task 2	0.912
Word embedding + RNN	Sub task 2	0.931

Table 5: Statistics of test results

Method	**Task**	**TP**	**TN**	**FP**	**FN**
Word embedding + CNN	Sub task 1	3479	237	238	346
Word embedding + RNN	Sub task 1	3446	224	251	379
Word embedding + RNN	Sub task 2	3193	363	133	506

through CNN and RNN, it registered an overall improved score from the previous MLP model. Specifically, the CNN gave the highest score for sub task 1, whereas RNN gave the best score for sub task 2, over the validation set. The MLP model with 6 hidden layers of size 300 are used primarily for building the base model. The activation function is *ReLU* and the dropout is 0.01. Model is implemented in *Keras*, which used the best validation score among 500 epochs. Then the model structure is extended into CNN and RNN neural network models. CNN is implemented with 256 filters and maxpooling is used for dimensionality reduction between the dense layers. All experiments were performed on GPU enabled TensorFlow[ABC+16] in conjunction with the Keras framework[C+15]. All models are trained using backpropagation.

5 Conclusion

Phishing emails have always plagued even the average user and classifying the same properly is a challenging task. Where former machine learning techniques failed, deep learning models have provided state of the art performance. The CNN/RNN/MLP architecture along with the Word2vec embeddings used in this work has outperformed former rule based and machine learning based models. During training the model gave high accuracy, while the test accuracy were comparatively low due to the highly unbalance nature of the dataset. In the proposed system, no external data was provided to train the model. CNN had a slightly better performance over RNN model on subtask1 and RNN perform well for subtask2, on the test data. For subtask 1, the CNN managed a score of 95.2%, almost comparable to RNN and for subtask 2, the RNN managed a score of 93.1%, making the RNN a better and more versatile overall performer. More accuracy can be achieved with these trained model by extrapolating the training corpus and by adding deeper layers to infuse more feature learning capabilities to the model. This work also demonstrates the possibilities of amalgamating techniques from text analytics and deep learning for cybersecurity use cases.

Acknowledgements

This research was supported in part by Paramount Computer Systems. We are 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

[AAY11] Tiago A Almeida, Jurandy Almeida, and Akebo Yamakami. Spam filtering: how the dimensionality reduction affects the accuracy of naive bayes classifiers. *Journal of Internet Services and Applications*, 1(3):183–200, 2011.

[ABC+16] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016.

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

[BG17] Reshma U. Anand Kumar M. Soman K.P. Barathi Ganesh, H.B. Representation of target classes for text classification-amrita-cen-nlp@rusprofiling pan 2017. In *CEUR Workshop Proceedings*, pages 25–27, 2017.

[BG18] Anand Kumar M. Soman K.P. Barathi Ganesh, H.B. From vector space models to vector space models of semantics. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 10478 LNCS.*, pages 50–60, 2018.

[C$^+$15] François Chollet et al. Keras, 2015.

[CL12] Godwin Caruana and Maozhen Li. A survey of emerging approaches to spam filtering. *ACM Computing Surveys (CSUR)*, 44(2):9, 2012.

[CM01] Xavier Carreras and Lluis Marquez. Boosting trees for anti-spam email filtering. *arXiv preprint cs/0109015*, 2001.

[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.

[FRID$^+$07] Florentino Fdez-Riverola, Eva Lorenzo Iglesias, Fernando Díaz, José Ramon Méndez, and Juan M Corchado. Applying lazy learning algorithms to tackle concept drift in spam filtering. *Expert Systems with Applications*, 33(1):36–48, 2007.

[GGWM$^+$10] Pedro H Calais Guerra, Dorgival Guedes, J Wagner Meira, Cristine Hoepers, MHPC Chaves, and Klaus Steding-Jessen. Exploring the spam arms race to characterize spam evolution. In *Proceedings of the 7th Collaboration, Electronic messaging, Anti-Abuse and Spam Conference (CEAS), Redmond, WA*, 2010.

[HDC$^+$18] Reza Hassanpour, Erdogan Dogdu, Roya Choupani, Onur Goker, and Nazli Nazli. Phishing e-mail detection by using deep learning algorithms. In *Proceedings of the ACMSE 2018 Conference*, page 45. ACM, 2018.

[HMR$^+$86] Geoffrey E Hinton, James L McClelland, David E Rumelhart, et al. Distributed representations. *Parallel distributed processing: Explorations in the microstructure of cognition*, 1(3):77–109, 1986.

[KRA$^+$07] Ponnurangam Kumaraguru, Yong Rhee, Alessandro Acquisti, Lorrie Faith Cranor, Jason Hong, and Elizabeth Nunge. Protecting people from phishing: the design and evaluation of an embedded training email system. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 905–914. ACM, 2007.

[LF17] Ruidan Li and Errin W Fulp. Evolutionary approaches for resilient surveillance management. In *2017 IEEE Security and Privacy Workshops (SPW)*, pages 23–28. IEEE, 2017.

[LT04] Chih-Chin Lai and Ming-Chi Tsai. An empirical performance comparison of machine learning methods for spam e-mail categorization. In *Hybrid Intelligent Systems, 2004. HIS'04. Fourth International Conference on*, pages 44–48. IEEE, 2004.

[MBA18] Youness Mourtaji, Mohammed Bouhorma, and Daniyal Alghazzawi. New phishing hybrid detection framework. *Journal of Theoretical & Applied Information Technology*, 96(6), 2018.

[MG18] Ankur Mishra and BB Gupta. Intelligent phishing detection system using similarity matching algorithms. *International Journal of Information and Communication Technology*, 12(1-2):51–73, 2018.

[MSC$^+$13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[PHS18] Tianrui Peng, Ian Harris, and Yuki Sawa. Detecting phishing attacks using natural language processing and machine learning. In *Semantic Computing (ICSC), 2018 IEEE 12th International Conference on*, pages 300–301. IEEE, 2018.

[S$^+$09] Jyh-Jian Sheu et al. An efficient two-phase spam filtering method based on e-mails categorization. *IJ Network Security*, 9(1):34–43, 2009.

[SAZ18] Sami Smadi, Nauman Aslam, and Li Zhang. Detection of online phishing email using dynamic evolving neural network based on reinforcement learning. *Decision Support Systems*, 2018.

[SDHH98] Mehran Sahami, Susan Dumais, David Heckerman, and Eric Horvitz. A bayesian approach to filtering junk e-mail. In *Learning for Text Categorization: Papers from the 1998 workshop*, volume 62, pages 98–105, 1998.

[SKP18] Vysakh S Mohan Soman Kp, Vinayakumar R and Prabaharan Poornachandran. S.p.o.o.f net: Syntactic patterns for identification of ominous online factors. In *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE, [In-Press], 2018.

[SLH17] Doyen Sahoo, Chenghao Liu, and Steven CH Hoi. Malicious url detection using machine learning: A survey. *arXiv preprint arXiv:1701.07179*, 2017.

[Tre04] Konstantin Tretyakov. Machine learning techniques in spam filtering. In *Data Mining Problem-oriented Seminar, MTAT*, volume 3, pages 60–79, 2004.

[VH13] Rakesh Verma and Nabil Hossain. Semantic feature selection for text with application to phishing email detection. In *International Conference on Information Security and Cryptology*, pages 455–468. Springer, 2013.

[VSH12] Rakesh Verma, Narasimha Shashidhar, and Nabil Hossain. Detecting phishing emails the natural language way. In *European Symposium on Research in Computer Security*, pages 824–841. Springer, 2012.

[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.

[WC07] Chih-Chien Wang and Sheng-Yi Chen. Using header session messages to anti-spamming. *Computers & Security*, 26(5):381–390, 2007.

[WC11] John S Whissell and Charles LA Clarke. Clustering for semi-supervised spam filtering. In *Proceedings of the 8th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference*, pages 125–134. ACM, 2011.

[ZZY04] Le Zhang, Jingbo Zhu, and Tianshun Yao. An evaluation of statistical spam filtering techniques. *ACM Transactions on Asian Language Information Processing (TALIP)*, 3(4):243–269, 2004.