SpamClus: an agglomerative clustering algorithm for spam email campaigns detection

Daniel Díaz[1,2], **Wesam Al-Nabki**[3], **Laura Fernández-Robles**[1,2], **Enrique Alegre**[1,2], **Eduardo Fidalgo**[1,2], **Alicia Martínez-Mendoza**[1,2]
[1]Universidad de León, 24007, León, Spain
[2]Researcher at INCIBE (Spanish National Institute of Cybersecurity), 24005, León, Spain
[3]Machine Learning Engineer at Nymiz, 48001, Bilbao, Spain
[1]{ddiao, l.fernandez, enrique.alegre, eduardo.fidalgo, amartm}@unileon.es
[3]wesamalnabki@nymiz.com

Abstract

Spam emails constitute a significant proportion of emails received by users, and can result in financial losses or in the download of malware on the victim's device. Cyberattackers create spam campaigns to deliver spam messages on a large scale and benefit from the low economic investment and anonymity required to create the attacks. In addition to spam filters, raising awareness about active email scams is a relevant measure that helps mitigate the consequences of spam. Therefore, detecting campaigns becomes a relevant task in identifying and alerting the targets of spam. In this paper, we propose an unsupervised learning algorithm, SpamClus_1, an iterative algorithm that groups spam email campaigns using agglomerative clustering. The measures employed to determine the clusters are the minimum number of samples and minimum percentage of similarity within a cluster. By evaluating SpamClus_1 on a set of emails provided by the Spanish National Cybersecurity Institute (INCIBE), we found that the optimal values are 50 minimum samples and a minimum cosine similarity of 0.8. The clustering results show 19 spam datasets with 3048 spam samples out of 6702 emails from a range of three consecutive days and eight spam clusters with 870 spam samples out of 1469 emails from one day.

1 Introduction

In 2023, more than 45% of the emails received by individuals were spam (Kulikova et al., 2024) and this figure is projected to reach 4.48 billion emails per day by 2024 (Dixon, 2022). Due to the characteristics provided by emails, such as low economic investment and anonymity, spam campaign emails have become a useful tool employed by cyberattackers to perform criminal activities. Some examples are the advertising of fake products, scams that cause financial losses, the mass mailing of malware, or illegal activities that end, in many cases, in economic losses for companies or even individuals (Karim et al., 2021). To reduce the impact or potential damage to users, companies try to run awareness campaigns indicating actions to be avoided, such as downloading files from unknown email addresses. In addition, platforms like Gmail in turn implement filters that label emails as spam so that the user handles them with the greatest possible care, and thus avoids becoming a victim of malicious spam emails. Those filters use black and white lists of email addresses to identify spam, however, spammers are constantly developing techniques to bypass the spam filters of e-mail clients (Jáñez-Martino et al., 2023). These solutions are most of the time lagging due to spammers' ability to innovate and troubleshoot to bypass filters. Some of the techniques they use include hiding text, images, or even HTML tags in the email body. These elements may add noise when we try to cluster messages by their content (RAZA et al., 2021). Because of these reasons, it is important to understand and develop systems that can remove these spammers' tricks and classify effectively spam emails from legitimate ham emails. Clues for identifying spammers are usually hidden in multiple aspects such as content, behavior, relationships, and interaction with the review (Chen et al., 2018; Mewada and Dewang, 2023). Authors usually try to group emails by tagging them by subject. However, spam campaigns can contain a wide variety of topics and limiting the number of subjects can be an unrealistic scenario. In this paper, we propose the first version of an iterative algorithm that uses the agglomerative cluster to detect spam campaigns based on a minimum number of examples and a similarity percentage between emails from the same campaign, in this case measured as cosine similarity.

The rest of the paper is organized as follows: the literature review is presented in section 2. In section 3, we explain the methodology. Then, we introduce the experiments and results in section 4. Finally, the discussion, conclusions and future work are described in section 5 and section 6.

2 Literature review

Authors have applied different approaches to address spam campaign detection and spam clustering. Most of the works in this field use the content of the emails to group them and identify the campaigns. Li et al. (2013) followed this approach and applied a topic modeling technique based on Latent Dirichlet Allocation to detect spam reviews, but did not consider the identification of campaigns. However, authors such as Li and Hsieh (2006) used the URL as a basis to cluster spam campaigns using the amount of money mentioned in the email as an additional feature. Several email features have also been used for clustering phishing campaigns, as proposed by Althobaiti et al. (2023), who employed Mean Shift algorithm to group emails based on the email sender, subject, body, and URL. Dinh et al. (2015) also used several email features in their work, including the email content type, character set, subject, layout, URL, and attachment. Their proposal consisted of a software framework that identifies campaigns in real-time and labels and scores the campaigns detected. They employed a database to handle a large number of spam emails, a scoring mechanism to highlight severe spam campaigns and a visualization tool.

Typically, spam campaign detection is addressed as a binary classification problem. For example, Karim et al. (2021) proposed an unsupervised algorithm that clusters emails into ham and spam, based on the domain and header information. They used a dataset with 22,000 emails from several sources, such as Guenter (2021), TREC (NIST, 2007) and ENRON (2015) datasets. However, some authors manage this problem as a multi-classification problem, where they create clusters based on the topic of the spam campaign (Ligthart et al., 2021; Saidani et al., 2020). Wang et al. (2016) proposed a model based on auto-encoders and clustering algorithms for spam review detection, although they do not identify campaigns. Our approach addresses spam campaign detection as a clustering problem where we group our spam campaigns in clusters with the same topic without labeling the dataset. The samples of the clusters in most of the cases should be similar except for small differences in specific data such as personalized information.

Therefore, the objective of our research is not to detect spam or identify spam topics, but rather to identify campaigns, which consist of sets of emails with the same goal and similar characteristics, usually sent within a certain period of time. These campaigns often target users who have something in common, such as being clients of the same organization. It would be helpful for Computer Emergency Response Teams to detect if a campaign is taking place, allowing an early response to the attacks and would enable them to alert users.

While the most recent dataset for email clustering dates from 2019-2020 (Althobaiti et al., 2023), we use a set of emails provided by the Spanish National Cybersecurity Institute (INCIBE) consisting of spam emails from 2021. Additionally, in contrast with existing proposals by other authors who used techniques such as DBSCAN (Althobaiti et al., 2023) or topic modeling (Li et al., 2013), we propose a technique based on agglomerative clustering. Moreover, the goal of SpamClus is to group emails together if they are likely to belong to the same spam campaign, as opposed to other authors who aim to cluster emails by topic, without considering if they belong to a spam campaign.

In previous work, only Wang et al. (2016) have used an approach based on autoencoders. Transformer-based models have obtained promising performance in clustering tasks (Mehta et al., 2021) and they are able to provide information about context. Thus, regarding the input for the SpamClus algorithm, we use BERT embeddings instead of features.

Finally, regarding spammer tricks present in spam emails, only Saidani et al. (2020) considered the presence of such techniques by adding the recognition words with separate letters. In contrast, we add a pre-processing step that removes hidden text using OCR.

3 Methodology

3.1 Datasets and pre-processing

We used a set of 4829 spam emails provided by INCIBE of possible spam campaign emails. The dataset contains real English and Spanish emails collected in 2021. Analyzing the content of the emails, we found that several emails contain hidden text. This hidden text is not visible in email visors, and it is added by spammers to introduce noise and reduce the efficacy of spam filters, as hidden text contains random topic text (Jáñez-Martino et al., 2023). We used Optical Character Recognition (OCR) to remove the hidden text and extract only visible text from the emails. This pre-processing technique enables the extraction of the text that the user would see when receiving the email, and the removal of the random content that is unrelated to the spam campaign. In particular, we used the OCR technique provided by the python library Pytesseract to extract only the visible text from the email HTML image. The OCR pipeline first detects if the content contains HTML code and, in that case, takes a screenshot of the email body and then extracts the text using OCR. This approach assumes that all HTML emails contain hidden text.

Besides, we removed the special characters, the remaining HTML and CSS tags, and the query strings, and replaced the HTML quotes with the character itself (e.g &aacute is replaced by 'á').

We also noticed that spam campaigns might have information that changes depending on the person to whom the spammers send the email. This information is added to personalize the emails. To reduce differences between emails from the same campaigns, we replaced personal information included in emails, such as email addresses and even URLs, with tokens.

3.2 Iterative clustering algorithm

We propose a new algorithm named SpamClus 1. This algorithm assesses different values of the threshold in a decreasing manner, along with several iterations. Thus, it first forms large clusters and evaluates whether or not they form a spam campaign depending on the input arguments. To evaluate whether a cluster is considered a campaign, we computed the cosine similarity of each cluster. This is defined as the average of cosine similarities among all pairs of emails in the cluster.

The input arguments are the same set of data (emails), a minimum number of samples and a minimum value of similarity per cluster that must be met to be considered spam. First, it calculates an initial threshold. To calculate this threshold, we randomly take 300 samples of the dataset (no matter the topic of the sample) and calculate the Euclidean distance between them, and we take as the initial threshold the ceiling of the maximum distance between two of the 300 samples.

We also perform the pre-processing described in subsection 3.1 for the content of the emails. Next, we use a pre-trained BERT model to extract the embeddings from the preprocessed email content. We chose a BERT model with support for multiple languages because we have emails in English and Spanish.

After that, we start with the first iteration. Within each iteration, we use the embeddings to create clusters using agglomerative clustering and then we label each email with its respective cluster identification. The next step is to calculate the number of samples that are within each cluster and the cosine similarity of each cluster.

Based on the computed number of samples and the cosine similarity of each cluster, we consider the clusters that achieve higher values than the input arguments ($min\_samples$ and $min\_similarty$) as campaigns and save them in a new dataset. Moreover, we remove those clusters from the original dataset to disregard them in the next iteration. Finally, we decrease by one unit the threshold value and check the stop criteria. The stop criteria encompass two aspects: the threshold needs to remain positive and the number of samples of at least one cluster needs to be higher than the $min\_samples$ value. The threshold value is one of the input parameters for the agglomerative clustering algorithm and is calculated as the greatest possible distance between samples, therefore it needs to be a positive value. If the number of samples remaining for the current iteration is lower than $min\_samples$, or if the created clusters for this iteration do not reach this value, it can be concluded that the number of samples is insufficient to be considered a campaign.

4 Experiments and results

We aim at optimizing the number of clusters detected as spam campaigns while ensuring that these clusters do not mix emails of different topics. To this end, we fixed the input parameter, $min\_samples$ to 50 and calculate the $min\_similarty$ to 0.8. The $min\_samples$ was set due to the recommendation of a cybersecurity technician from INCIBE as the optimal number to detect a campaign. We computed the $min\_similarty$ as follows.

We ran SpamClus for several ranges of days taking from 14069 to 6702 samples and manually checked the content of the email clustered as spam campaigns. The objective was to identify the instances where the algorithm clusters unrelated campaigns, with the aim of maximizing the number of samples clustered as spam campaigns but avoiding clusters of mixed topics.

SpamClus 1 Spam detection algorithmic based on agglomerative clustering

```
$min\_Samples$
$min\_similarity$
$emails\_df$
$spam\_df \leftarrow DataFrame[\emptyset]$
$initial\_threshold \leftarrow compute\_threshold(df[content])$
$emails\_content \leftarrow preprocess\_content(df[content])$
$embeddings \leftarrow BERT\_Encode(emails\_content)$

$threshold \leftarrow initial\_threshold$
do
    $emails\_df[cluster] \leftarrow agglomerative\_clustering(embeddings, threshold)$
    $number\_samples \leftarrow count\_samples\_per\_cluster(emails\_df[cluster])$
    $cosine\_similarities \leftarrow cosine\_similarity\_per\_cluster(emails\_df[cluster])$

    $spam\_df \leftarrow spam\_df.append(emails\_df\ where(number\_samples > min\_Samples\ \&$
$cosine\_similarities > min\_similarity)$

    $emails\_df \leftarrow emails\_df\ where(number\_samples > min\_Samples\ \&$
$cosine\_similarities > min\_similarity))$

    $number\_samples \leftarrow count\_samples\_per\_cluster(emails\_df[cluster])$
    $threshold \leftarrow threshold - 1$
    $embeddings \leftarrow emails\_df[cluster]$
while $(threshold > 0\ \&\ any(number\_samples > min\_Samples))$

return $spam\_df, emails\_df$ ▷ Return spam campaigns clustered and emails no considered campaigns
```

Table 1 and Table 2 show the results obtained for the ranges for three and one day we used to compute an optimal *min_similarty*. Table 2 shows an optimal *min_similarty* in 0.6 since at this point the created clusters do not merge different topics in the same cluster. However, Table 1 shows that for a three-day range the optimal *min_similarty* is 0.8. Finally, we noticed that the value for *min_similarty* depends on the input emails, nevertheless, we set the value to 0.8 because it is the most frequent optimal value for the emails we tested. The column "Mixed clusters?" indicates whether the clusters created with the algorithm contain samples regarding different spam topics.

Min similarity	# Non-Spam Samples	# Spam Samples	# Spam Clusters	Mixed clusters?
0.9	3455	3247	16	No
0.8	**3654**	**3048**	**19**	**No**
0.7	4791	1911	22	Yes
0.6	5093	1609	24	Yes
0.5	5093	1609	24	Yes

Table 1: Spam clusters with different minimum cosine similarity (From 11/12/2021 to 13/12/2021)

5 Discussion

In addition to the aforementioned techniques, other methodologies were also evaluated. Initially, a fixed-threshold approach was employed, yet encountered difficulties with each threshold utilized. When we used a very high threshold, the small campaigns began to mix, leading to a high number of emails per cluster, in which sometimes emails from different topics are mixed. When the threshold is low, the algorithm creates many clusters with very high cosine similarity. This is because only very similar emails are clustered together, but with too few samples to be considered spam campaigns. Finally, to avoid the problem described above, we propose the Algorithm SpamClus 1.

With regard to the input parameters minimum number of samples and minimum similarity, it can be anticipated that we encounter similar issues

Min similarity	# Non-Spam Samples	# Spam Samples	# Spam Clusters	Mixed clusters?
0.9	261	1208	4	No
0.8	423	986	8	No
0.7	599	870	8	No
0.6	**599**	**870**	**8**	**No**
0.5	823	646	9	Yes

Table 2: Spam clusters with different minimum cosine similarity for one day (11/12/2021)

to those associated with modifying the threshold value. Decreasing the minimum number of samples would mean that it would be possible to create smaller clusters. However, because campaigns are created when spam emails are distributed on a large scale, we need to establish a minimum so that the amount is big enough to be considered a campaign. Increasing the minimum number of samples could cause campaigns to go undetected if the value is too high. As indicated in section 4, decreasing the value of minimum similarity increases the possibility of creating mixed clusters where samples belong to different topics. However, increasing the value too much could result in campaigns going undetected. In our work, the minimum number of samples was established by an INCIBE cybersecurity technician with experience in the field of spam campaigns. The value could be modified for other applications if considered appropriate, but the consequences mentioned above should be taken into account. Automating the selection of minimum similarity is proposed as future work.

6 Conclusions and future work

In this work, we presented a baseline algorithm that addressed the problem of spam campaign detection using agglomerative clustering named SpamClus 1. This algorithm avoids having a fixed threshold, which creates small clusters with high similarity or big clusters with low similarity. The output of this algorithm is two variables that contain the clusters considered spam and non-spam. This output depends on two criteria to consider a campaign: the minimum number of samples and the similarity of the emails in the cluster. We fixed the first parameter to 50 based on experts' recommendations and calculated the second to 0.8 depending on the most frequent optimal value tested on several ranges of dates emails. With those fixed values, we obtained 19 spam clusters with a total of 3048 samples for a range of three days (see Table 1) and 8 spam clusters with a total of 870 samples for one day (see Table 2).

In future work, we will explore new approaches to automatically compute the minimum cosine similarity depending on the emails being clustered. In addition, we might explore new options to remove hidden text from the email content because the OCR approach takes too long to extract only visible text.

Furthermore, the current proposal to remove hidden text from emails using OCR requires a significant computational cost. Therefore, in future work, we propose to explore alternative methods for removing hidden text and preserving only the relevant email content.

Limitations

In this work, we have evaluated SpamClus 1 algorithm using five different minimum cosine similarity values, ranging from 0.5 to 0.9. Our findings indicate that the optimal value is 0.8. The algorithm could be improved by incorporating an automatic calculation of the similarity value, based on the emails being clustered.

Ethical statement

This work can contribute to **society and human well-being** and **avoid harm**: by ensuring the safety and security of individuals and organizations who may otherwise fall victim to cyber threats. The **system** to detect spam email campaigns can contribute to alerting individuals and companies targeted by spam campaigns and reducing the number of victims of spam attacks.

Use of AI Technologies: We recognize the potential for misuse of AI technologies, including the possibility of adversarial attacks. We advocate for the ethical use of AI in cybersecurity, emphasizing its role in protecting individuals, organizations, and societies against cyber threats.

Acknowledgements

This work has been funded by the Recovery, Transformation, and Resilience Plan, financed by the European Union (Next Generation), thanks to the LUCIA project (Fight against Cybercrime by applying Artificial Intelligence) granted by INCIBE to the University of León.

References

Kholoud Althobaiti, Kami Vaniea, Maria K Wolters, and Nawal Alsufyani. 2023. Using clustering algorithms to automatically identify phishing campaigns. *IEEE Access*.

Hao Chen, Jun Liu, Yanzhang Lv, Max Haifei Li, Mengyue Liu, and Qinghua Zheng. 2018. Semi-supervised clue fusion for spammer detection in sina weibo. *Information Fusion*, 44:22–32.

Son Dinh, Taher Azeb, Francis Fortin, Djedjiga Mouheb, and Mourad Debbabi. 2015. Spam campaign detection, analysis, and investigation. *Digital Investigation*, 12:S12–S21. DFRWS 2015 Europe.

S. Dixon. 2022. Number of e-mail users worldwide 2025.

ENRON. 2015. ENRON Email Corpus.

B. Guenter. 2021. Spam collection.

Francisco Jáñez-Martino, Rocío Alaiz-Rodríguez, Víctor González-Castro, Eduardo Fidalgo, and Enrique Alegre. 2023. A review of spam email detection: analysis of spammer strategies and the dataset shift problem. *Artificial Intelligence Review*, 56(2):1145–1173.

Asif Karim, Sami Azam, Bharanidharan Shanmugam, and Krishnan Kannoorpatti. 2021. An unsupervised approach for content-based clustering of emails into spam and ham through multiangular feature formulation. *IEEE Access*, 9:135186–135209.

T. Kulikova, O. Svistunova, A. Kovtun, I. Shimko, and R. Dedenok. 2024. Spam and phishing in 2023.

Fulu Li and Mo-Han Hsieh. 2006. An empirical study of clustering behavior of spammers and group-based anti-spam strategies. In *CEAS 2006 - The Third Conference on Email and Anti-Spam, July 27-28, 2006, Mountain View, California, USA*, volume 2006, pages 21–28.

Jiwei Li, Claire Cardie, and Sujian Li. 2013. Topicspam: a topic-model based approach for spam detection. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 217–221.

Alexander Ligthart, Cagatay Catal, and Bedir Tekinerdogan. 2021. Analyzing the effectiveness of semi-supervised learning approaches for opinion spam classification. *Applied Soft Computing*, 101:107023.

Vivek Mehta, Seema Bawa, and Jasmeet Singh. 2021. Weclustering: word embeddings based text clustering technique for large datasets. *Complex & intelligent systems*, 7(6):3211–3224.

Arvind Mewada and Rupesh Kumar Dewang. 2023. A comprehensive survey of various methods in opinion spam detection. *Multimedia Tools and Applications*, 82(9):13199–13239.

TREC NIST. 2007. TREC Spam Collection.

Mansoor RAZA, Nathali Dilshani Jayasinghe, and Muhana Magboul Ali Muslam. 2021. A comprehensive review on email spam classification using machine learning algorithms. In *2021 International Conference on Information Networking (ICOIN)*, pages 327–332.

Nadjate Saidani, Kamel Adi, and Mohand Said Allili. 2020. A semantic-based classification approach for an enhanced spam detection. *Computers & Security*, 94:101716.

Baohua Wang, Junlian Huang, Haihong Zheng, and Hui Wu. 2016. Semi-supervised recursive autoencoders for social review spam detection. In *2016 12th International Conference on Computational Intelligence and Security (CIS)*, pages 116–119. IEEE.