Context-based Clustering to Mitigate Phishing Attacks

Tarini Saka
s2138664@ed.ac.uk
University of Edinburgh
Edinburgh, United Kingdom

Kami Vaniea
kvaniea@inf.ed.ac.uk
University of Edinburgh
Edinburgh, United Kingdom

Nadin Kökciyan
nadin.kokciyan@ed.ac.uk
University of Edinburgh
Edinburgh, United Kingdom

ABSTRACT

Phishing is by far the most common and disruptive type of cyber-attack faced by most organizations. Phishing messages may share common attributes such as the same or similar subject lines, the same sending infrastructure, similar URLs with certain parts slightly varied, and so on. Attackers use such strategies to evade sophisticated email filters, increasing the difficulty for computing support teams to identify and block all incoming emails during a phishing attack. Limited work has been done on grouping human-reported phishing emails, based on the underlying scam, to help the computing support teams better defend organizations from phishing attacks. In this paper, we explore the feasibility of using unsupervised clustering techniques to group emails into scams that could ideally be addressed together. We use a combination of contextual and semantic features extracted from emails and perform a comparative study on three clustering algorithms with varying feature sets. We use a range of internal and external validation methods to evaluate the clustering results on real-world email datasets. Our results show that unsupervised clustering is a promising approach for scam identification and grouping, and analyzing reported phishing emails is an effective way of mitigating phishing attacks and utilizing the human perspective.

CCS CONCEPTS

• **Security and privacy** → **Phishing**; • **Computing methodologies** → **Cluster analysis**; **Information extraction**.

KEYWORDS

Phishing, Clustering, Context, Artificial Intelligence, Human-Centered Artificial Intelligence

ACM Reference Format:

Tarini Saka, Kami Vaniea, and Nadin Kökciyan. 2022. Context-based Clustering to Mitigate Phishing Attacks. In *Proceedings of the 15th ACM Workshop on Artificial Intelligence and Security (AISec '22), November 11, 2022, Los Angeles, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3560830.3563728

1 INTRODUCTION

Phishing emails are malicious emails that often result in a large amount of financial and reputational damage to organizations. Therefore, employees are regularly told to report phishing to their organizations' computing support desk or Information Security (IS) teams. However, processing these reports at scale is quite challenging [3]. Problems range from the number of reports to the fact that phishing is specifically designed by attackers to bypass automated filters, making it challenging to automatically extract reliable features. In this work, we aim to better support the people who read phishing reports by clustering the reports such that they can be read and replied to as a unit rather than as individual reports, and hence help fight phishing attacks in a timely manner.

Phishing works by sending people fraudulent emails, usually pretending to be from a well-known source, with the intent of tricking recipients into giving up sensitive information, such as passwords or credit card numbers. In the first quarter of 2022, the Anti-Phishing Working Group (APWG) observed over a million phishing attacks, which is the worst quarter they have ever observed [6]. According to IBM's Cost of a Data Breach Report 2021 [52], it was found that 2021 was the costliest year for data breaches in the last 17 years. They reported that from 2020 to 2021, the average total cost of a data breach increased from $3.86 million to $4.24 million. Researchers from the same report found that, on average, data breaches caused by phishing took 213 days to be identified and 80 days to be contained. Meaning that the average time it takes to contain a phishing threat overall is 290 days. According to a survey conducted by the email security provider IRONSCALES, 81% of organizations around the world have experienced an increase in email phishing attacks since March 2020, with a particular increase in COVID-19 related phishing attacks. Popular themes include stimulus checks, fake CDC warnings, Netflix scams, fines for coming out of quarantine and many more. According to their research, since the onset of the COVID-19 pandemic, more than one-third of IT professionals spend all of their time remedying phishing attacks, and 74% spend more than 30 minutes addressing each attack [30].

Given the risks and expenses involved, organizations go to great lengths to prevent phishing from being successful, best practice recommends a mix of proactive security measures like automated filters on email servers and reactive measures which include encouraging employees to report any phishing that they encounter [3, 17, 63]. While automated detection of phishing is impressively effective, with some algorithms achieving accuracy rates over 99% [22, 25], it is still the case that some phishing makes it through into user inboxes. To identify and block these cases, best practice recommends getting employees to report phishing that they see and then use the reports to remove phishing from inboxes and update protections like filters. This approach is very effective, but it still involves humans looking through each phishing report. In an organization of 50,000, if even 10% of people report a phishing email that is 500 reports that must be examined by a human to judge if it i) is actually phishing and ii) is new phishing or phishing that has already been

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
AISec '22, November 11, 2022, Los Angeles, CA, USA
© 2022 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-9880-0/22/11…$15.00
https://doi.org/10.1145/3560830.3563728

handled (e.g. does anything need to be done?). It is also necessary to respond back to the person who sent the report which also takes time.

One potential solution to this issue is to build a system that is capable of clustering together phishing reports that are similar enough that a single response could be written and sent back to everyone who reported an email in the cluster. For example, an attacker might send many emails claiming to be a common parcel delivery service that just needs a fee to deliver the package. Even though the emails sent may be quite different in content and even mimic different parcel delivery services, a single response could be sent to everyone who reports these emails explaining that it is a parcel scam and should be deleted. In this work, we explore the best way to cluster reported phishing into scams. That is, emails that are all using a similar scam to trick users, even if the emails themselves are sent by more than one attacker.

Our approach aims to utilize existing machine learning approaches to analyze and group reported phishing emails so as to help organizations take immediate action against attacks. In this paper, we introduce an intelligent system capable of clustering reports such that a helpdesk staff could review each cluster of reported phish and provide a single useful response to all the people who reported emails in the cluster.

To accomplish this, we explore the feasibility of using unsupervised clustering techniques to identify and group together phishing emails of similar scams. That is, phishing emails that share common attributes like the underlying fraud, the organization being impersonated, the response provoked, and so on. For example, banking-related phishing scams, file transfer scams and so on. We further utilize transformers in our algorithm to capture the context better. Using word-embedding techniques to represent the textual features of phishing emails is a relatively new approach. The combination of word-embeddings and unsupervised clustering for the purpose of identifying the underlying scam in an email, makes our work rather unique and novel. The experimental results, which were based on numerous internal and external evaluation metrics (ARI, AMI, V-Measure, Purity, Silhouette Score, and DB-Index), showed that K-Means clustering shows a relatively better overall performance and the best clustering combination was observed to be Agglomerative Clustering with Feature Set 4 comprising the BERT vectors for the email body text and the subject-line topic models. Our contributions in this paper are as follows:

(1) We employ word-embedding techniques to extract contextual features from phishing emails;
(2) We define seven distinct feature sets incorporating different parts of the email;
(3) We implement unsupervised clustering algorithms to group together phishing emails of similar scams;
(4) We evaluate the proposed comparative approach to identify the feature set that best captures email context by using real-world email datasets.

The remainder of this paper is organized as follows. Section 2 elaborates on recent previous related works that focus on phishing email clustering, scam and campaign detection, and email context representation. Section 3 details the methodology of the proposed approach. Section 4 evaluates the clustering performance and discusses the results of the experiments. Finally, section 5 summarises the critical points and outlines the conclusions drawn.

2 RELATED WORK

Phishing is by far the most common and disruptive type of cyber-attack faced by most organizations [17, 58]. While the damage caused on its own may not seem to be large, it is often used as a gateway attack to gain creditials which are then used launch more damaging attacks such as ransomware or privilege escalation. Highly damaging attacks such as the power station shutdown in the Ukraine [66] often start with a phishing email. Given the low costs and high potential rewards, phishing is a common attack strategy and it is only getting more popular.

Extensive research has gone into detecting phishing attacks, preferably before they reach end users [8, 10, 33]. The area is especially hard to solve because of the existence of an active adversary who adjusts to each new defense. For example, automated filters are now used on most mail servers, but these filters need to work fast and therefore cannot easily do deep analysis of URLs or attachments. Attackers make use of this, so sandboxing [62] was introduced where an email that contains unknown URL links, file types, or suspicious senders is be tested in a secure space before it reaches the network or mail server. On the user's end, browsers can have blocklists added to prevent the user from visiting known phishing sites even if complex URL redirection is used [9, 35]. These automated solutions are very effective, but the existence of an active attacker means that some phishing will always get through, requiring that users be involved in detection. A best practice here is to ask users to report suspicious looking email which is then reviewed by help desk staff or a security team [29, 34]. The major problem with this approach is scale and time [3, 57]. If many users report emails then it takes time to manually review them, which means a slower response time to the attack and a slower response to the people reporting.

One way of addressing the challenge of manual review overload could be automated grouping of similar reported emails. Past research on email grouping is fairly limited and has been mostly used for managing email overload by creating email groups [53, 64] or to classify incoming emails into relevant groups like personal/professional [2]. Some works focus on unsupervised machine learning techniques, which are relevant to our work. In the cybersecurity field, email grouping has been mostly focused on clustering emails into scam/benign categories [15, 32]. DeBarr et al. [15] utilise Spectral Clustering to analyze the links between URL substrings for websites found in the email contents and then use a Random Forest classifier for phishing identification. Karim et al. [32] present an in-depth analysis on using unsupervised clustering of only the domain and header information to classify emails into spam or ham (benign). There has also been research on profiling phishing as a clustering problem to determine the activity of an individual or a particular group of phishers [27, 28]. While similar to our work, they focus more on using phishing profiles to understand phishers better and predict their behaviour.

Some work has been carried out on the use of unsupervised clustering to identify phishing scams or campaigns. Alazab et al. [1]

introduce the idea of 'Authorship Attribution' for spam campaign identification. They use an Unsupervised Automated Natural Cluster Ensemble (NUANCE) model to separate spam emails into groups with similar topics. They use 57 stylistic and structural features including the total count of the: words in the text of the email, punctuation used in the email body, contractions present in the email, URLs present in the body of the email, and obfuscated words present in the email. In [19], Dinh et al. propose a software framework for spam campaign detection, analysis and investigation. They use a frequent-pattern tree on a combination of header and content features to identify spam campaigns on-the-fly. Other similar works include spam email clustering based on email content [26, 61]. Although these approaches provides good results in spam campaign detection, they do not take into consideration any contextual information, such as word embeddings, from the emails, as we do in this work. The features used in these works are better suited for campaign detection, where the threat origin (or author) is common, and hence contextual information may not be required. But our work aims to identify the underlying scams in emails, irrespective of the origin, and hence context is more relevant.

With the advancements in Natural Language Processing (NLP) and word embedding techniques, contextual and semantic information can be extracted from email text, and used in machine learning tasks such as classification and clustering [50, 59]. Bountakas, Koutroumpouchos and Xenakis [11] introduce an approach where they combine NLP techniques together with machine learning methods for the detection of phishing emails. They use word embedding techniques such as TF-IDF [46], Word2Vec [38], and BERT [18] to represent the email text and apply a variety of classification algorithms to identify phish and benign emails. They were the first to use the BERT transformer model to extract textual features from phishing emails. They identified Word2Vec + Random Forest algorithm as the best combination for a balanced dataset and show that word-embedding techniques can achieve state-of-the-art results in the detection of phishing attacks. Another similar approach is proposed by Somesha and Pais [56], combines six word embedding techniques with five machine learning classifiers to evaluate the best performing combination. These works show that word embeddings capture essential contextual information and provide an efficient representation of email text. They also suggest that word embeddings algorithms could be combined with unsupervised machine learning techniques, as we show in this paper.

3 METHODOLOGY

Our proposed methodological approach consists of five sub-tasks: (i) Email Parsing and Pre-processing, (ii) Feature Extraction, (iii) Feature Representation, (iv) Clustering, and (v) Cluster Evaluation. We implement three different clustering algorithms (K-Means, DBSCAN, and Agglomerative) on seven different feature sets, and use a variety of clustering evaluation methods to identify the best approach to work with to achieve our goals. Figure 1 provides an outline of the proposed approach and the various sub-tasks performed. Our implementation is freely available online[1]. We now describe each of the components in detail.

[1]https://git.ecdf.ed.ac.uk/s2138664/phishing-research

3.1 Email Parsing and Pre-processing

3.1.1 Dataset. For the initial development of the algorithm, we use both phishing and a benign datasets. There is always a possibility of benign emails getting reported to the help desk and our algorithm should be able to differentiate between phishing and benign emails. Hence, we use the benign email set as validation. The phishing corpus consists of 2279 emails taken from the Nazario PhishingCorpus [39], a publicly available collection of hand-screened phishing messages collected between February 2000 to July 2007. The benign emails are taken from the publicly-available Enron Email dataset developed by the CALO Project [12]. The Enron corpus is a large dataset containing around 500,000 emails that correspond to the day-to-day workplace interactions between employees of the Enron Corporation. At the time of this project, the Nazario phishing corpus link was not valid and hence we obtained the dataset from a secondary source, which is accessible through their GitHub repository [41]. Both the phishing and benign sets were downloaded from this source, each in the form of a mbox, a plain text file of concatenated email messages. The mbox contained all the emails in a single file and was read on Python 3 using the `mailbox` module.

3.1.2 Email Parsing and Pre-processing. We process the dataset to extract relevant information from different parts of an email. From the email header, we extract: *Message-ID*, *Multipart (yes/no)*, *From Address*, *Date*, *Return Path*, *Reply-To Path*, and the *Subject-Line*. From the email body, we extract the text content, words, and URLs. We exclude any emails that have empty content, and the remaining dataset contained 2193 phishing emails. Due to limited work in reported phishing, we do not know what phish/benign ratios appear in the wild, so we use a balanced set. To create a balanced set, we take an appropriate subset of 2193 emails from the Enron corpus. We follow the standard data pre-processing steps; the email text and subject line are converted into lowercase, and the special characters, stopwords, and punctuation marks are removed. Each email is then labelled with a phish/benign tag to be used in the post-clustering evaluation.

3.1.3 Manually Identified Scams. To compare the different clustering algorithms and evaluate their ability to identify and separate scams, we use a small tagged subset of the phishing dataset. To create this tagged subset, the lead author reviewed the phishing dataset and identified key phrases found in the most common scams. These were further discussed and analysed by all authors resulting in 14 scams that could be identified this way. The lead author then ran a comparison for each key phrase against the whole dataset, manually reviewed all the emails with similar content and tagged them together by scam with a unique number. Using the 14 key phrases identified by the authors, we tagged 493 emails (22.48% of the phishing dataset). The tagged dataset size was selected to represent about a quarter of the dataset, particularly the most common scams which would ideally be located together in clusters. The summary of the manually identified and tagged scams is provided in Table 1. The first column of the table is the unique identifying number or tag assigned to each scam, the second column is the corresponding key phrase, and the third states the characteristic features of the underlying scam.

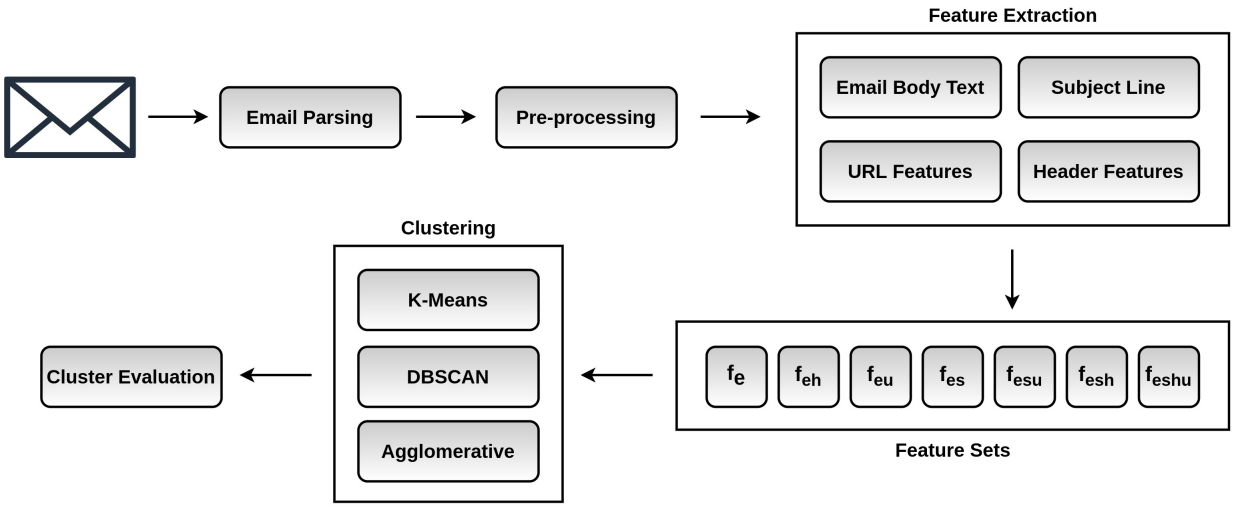


Figure 1: Outline of the proposed approach and the sub-tasks. The pipeline starts with Email Parsing and Pre-processing, followed by Feature Extraction and Representation, then seven Feature Sets are defined, followed by clustering with three algorithms and finally, post-clustering analysis of the clusters formed.

Table 1: Summary of the key phrases and characteristic features used to manually label scams in the dataset

Tag (# of emails)	Key phrase	Characteristics
Tag 1 (29)	"limit sensitive account"	PayPal Account ; lost/stolen card ; Different Case ID
Tag 2 (19)	"external secured server"	Bank of America ; system maintenance/upgrade
Tag 3 (82)	"foreign IP address"	Various organization ; login from foreign IP address ; spoofed links
Tag 4 (160)	"Question about Item"	eBay ; message from buyer/seller ; spoofed links
Tag 5 (65)	"quickly restore full access"	PayPal, eBay ; unusual account activity
Tag 6 (19)	"confirm your banking details"	PayPal, Bank of America ; account from different locations
Tag 7 (15)	"New messages in My Messages"	eBay ; new message in eBay account ; spoofed link
Tag 8 (13)	"confirm your Account Informations"	eBay ; security changes ; spoofed links
Tag 9 (17)	"additional email address"	PayPal ; new email Id added ; spoofed links ; different case ID
Tag 10 (10)	"series of verification process"	PayPal ; random verification
Tag 11 (11)	"Federal Credit Union network"	Federal Credit Union ; account update ; spoofed link
Tag 12 (9)	"upgrade our servers"	eBay, PayPal ; account blocked
Tag 13 (37)	"safety and integrity"	Various organizations ; safety concerns ; spoofed links
Tag 14 (7)	"security service notification"	eBay ; account suspended ; very threatening tone

3.2 Email Representation with Feature Sets

Based on previous phishing research [20, 23], and our observations of phishing messages, we identify the following features as relevant and important to represent an email. Table 2 provides a summary of the different email features considered, the corresponding feature representations, and vector size.

Email Body. Here, we focus on the features that could be extracted from the email body such as the number of parts in the email and their type, and the presence and number of URLs. An important feature extracted from this category is the email text which contains information regarding the context and purpose of an email, which is essential to identify similar scams. For this, we extract the plain text from the email body and use word embedding techniques to represent it. We initially used TF-IDF (term frequency-inverse document frequency) [45, 46], a traditional text vectorizing algorithm that quantifies the importance of string representations

(words) in a document amongst a collection of documents. In a preliminary study, we experimented with TF-IDF vectors but it showed poor results. TF-IDF cannot represent the semantic meaning or context of the text, and it suffers from memory inefficiency and the curse of dimensionality. We then consider other advanced word-embedding techniques to represent the email body text. In particular, we choose BERT (Bidirectional Encoder Representations from Transformers) [18], a transformer-based model to convert text into vectors, as it can capture contextual information in text and can adapt well to new data. In particular, we use the BERT 'multi-qa-MiniLM-L6-cos-v1' model that maps the text to a 384 dimensional dense vector space and was designed for semantic search.

Subject Line. This short piece of text is the first thing a user sees when they get an email. Cybercriminals design subject lines in a way that creates urgency, personalisation and pressure in order to trick victims into clicking on malicious links, downloading malware, and so on [43]. To represent the subject line semantically, we generate topic models from the data and use a bag-of-topics approach. Following the approach proposed in [55], we cluster the BERT vectors of trigrams generated from the phishing dataset subject lines to generate 30 topic clusters and extract the top 20 keywords, by frequency, for each topic. We then count the number of keywords corresponding to each topic present in the subject line and generate topic vectors of dimension 30. More details regarding the subject model can be found in Appendix A.

URL Features. URLs present in phishing emails are an important criterion to detect or classify phishing attacks [7]. The URL destinations are aspects that phishers cannot fully hide or manipulate, and hence provide vital information regarding the phishing attack [4]. Features extracted from the URLs are: the number of URLs, average length of the URLs, the maximum number of dots in all the URLs, number of unique domains, and domain check with the top 10000 domains in the Tranco list [44]. The Tranco list was chosen because it incorporates several popularity metrics and is designed to be used in research. It averages out rankings over thirty days instead of relying on a snapshot of the list from just one day.

Header Features. The email header contains information about the sender, recipient, the email's route between servers and various authentication details. Source-oriented features extracted are the 'From' address, 'Reply-To' address, and 'Return-Path' address. Spoofing the 'From' address so that it seems legitimate is a common trick phishers use. To accommodate this, we compare the 'From' address domain to the 'Return-Path' and 'Reply-To' domain. We also compare it to all URL domains. Other features extracted from the header are the number of attachments. Another important header feature is the DKIM signature [59], which was not present in the datasets and hence not considered.

In Table 3, we outline the seven different combinations of features used as input for the clustering algorithms. Each feature set is labelled as f_x, where x refers to specific features being used (e.g. h to represent header features). We also define an additional feature set f_b, without any context-based features, to use as a reference baseline. This consists of only the header and URL features.

3.3 Clustering Algorithms for Grouping Emails

Many different types of clustering methods have been proposed in the literature [47] such as K-Means [36], DBSCAN [21, 51], Hierarchical [40], BIRCH [68], OPTICS [5] and Spectral clustering [54]. Choosing the right clustering algorithm requires taking into consideration a lot of factors. Firstly, we need a clustering technique that can accommodate high-dimensional data. The algorithm should also be able to deal with outliers such as single emails, non-phishing emails and so on. Since the number of scams is not pre-determined, the selected clustering algorithm should either automatically decide on the cluster number or we need to define a method to get the optimal number. We identified three candidate algorithms, namely K-means, DBSCAN, and Hierarchical Agglomerative.

K-Means [36] is a centroid-based algorithm that identifies k number of centroids in the dataset and then allocates every data point to the nearest cluster. The main objective of this algorithm is to minimize the sum of distances (Euclidean distances) between the points and their respective cluster centroid. It is computationally very fast and hence suitable for large datasets. The main limitation of this algorithm is the requirement of the number of clusters as an input parameter; which can be computed by the Elbow Method [31] and the Silhouette Method [49]. We initialize the K-Means algorithm with the the number of clusters provided by these methods, but the performance was observed to be sub-optimal with respect to their ability to identify and separate scams.

Density-Based Spatial Clustering of Applications with Noise (*DBSCAN*) [21, 51] involves finding high-density areas in the data and expanding around them. The algorithm takes two inputs: the maximum distance between two samples for one to be considered as in the neighbourhood of the other (eps) and the number of samples in a neighbourhood for a point to be considered as a core point (n). It generates clusters by determining core data points with a neighbourhood that includes at least 'n' points less than 'eps' distance away from the core. It estimates the number of clusters from the dataset automatically. If a point is not within the eps-neighbourhood of any core, then it is labelled as noise. The main drawback is the algorithm does not cope well with datasets where clusters have varying similarity levels and sizes.

Agglomerative clustering [40] is a type of hierarchical clustering, where each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy. Similar to K-Means, the main limitation of this algorithm is the requirement of the number of clusters as an input parameter. Generally, this value is learned by plotting the dendrogram of the cluster hierarchy.

3.3.1 Hyper-parameter Optimization: Tuning a Machine Learning model is a type of optimization problem. Each model has a set of hyper-parameters and we aim to find the right combination of their values to either minimize or maximize a function. This step is particularly important to compare the performance of different Machine Learning models on a dataset. We optimize the different clustering algorithms by maximizing their accuracy with respect to the validation metrics defined in 3.5. The hyper-parameter optimizations is done using a two-step process: *Step 1:* We run a coarse-grained parameter sweep over a large range of input values and plot an accuracy vs input graph. *Step 2:* Using the graphs generated in Step

Table 2: The representation of features selected

Feature Type	Feature Representation	Vector Size
Email Body text (f_e)	BERT word embeddings	384
Subject line features (f_s)	BERT-based topic model	30
Header features (f_h)	presence of 'Reply-To' and 'Return-Path'; 'From' and 'Return-Path' domain match; 'From' and 'Reply-To' domain match; number of attachments; Tranco domain check; URL domains and 'From' domain match	7
URL features (f_u)	number of URLs; average length of the URLs; maximum number of dots; number of unique domains; Tranco domain check	6

Table 3: The identified feature sets to represent emails

Feature Set	Components	Vector Size
f_e	Email Body Text	384
f_{eh}	Email Body Text + Header features	391
f_{eu}	Email Body Text + URL features	390
f_{es}	Email Body Text + Subject line features	414
f_{esu}	Email Body Text + Subject line features + URL features	420
f_{esh}	Email Body Text + Subject line features + Header features	421
f_{eshu}	Email Body Text + Subject line features + Header features + URL features	427

1, we run a fine-grained sweep around the peaks and identify the optimal input.

For K-Means and Agglomerative clustering, we run a parameter sweep of the number of clusters and for DBSCAN, we run a parameter sweep of the epsilon value.

3.4 Feature Selection

Feature selection is the process of extracting a subset of the most relevant features to improve the performance of Machine Learning models in terms of accuracy and training time [24]. We utilise Chi-Square feature selection [67], as it has achieved promising results in previous works and is easy to implement [65]. Chi-Square is a statistical hypothesis test, which measures the relationship between two variables and identifies the level of correlation. We use Chi-Square feature selection on all the features extracted to measure the correlation of each feature with the email's phish/benign tag. We then run clustering with the identified Chi-Square subset.

3.5 Clustering Evaluation

The ideal clustering algorithm should: (i) group together emails of similar scams and also be able to differentiate any benign emails reported for being suspicious; (ii) produce homogeneous clusters, where all emails in one cluster are either benign or belong to the same scam. In this section, we explain the various validation methods used to evaluate the clustering performance of the chosen algorithms according to our criteria.

3.5.1 Internal Evaluation: Internal measures evaluate how well the clusters are formed with respect to their compactness and separation [16]. They usually assign the best score to the algorithm that produces clusters with high similarity within a cluster and low similarity between clusters. These measures do not require any prior cluster labelling or ground-truths.

Silhouette Coefficient Score [49]: The Silhouette Coefficient for each sample is calculated using two measures: (i) the mean distance between a sample and all other points in the same class, and (ii) the mean distance between a sample and all other points in the next nearest cluster. The Silhouette Score for the whole dataset is given as the mean of the Silhouette Coefficient for each sample. The score ranges between -1 and +1, with values around zero indicating overlapping clusters. The score is higher when clusters are dense and well separated.

Davies–Bouldin Index [13]: The index is defined as the average similarity of each cluster with its most similar cluster, where similarity is the ratio of within-cluster distances to between-cluster distances. Thus, clusters which are farther apart and less dispersed will result in a better score. The minimum score is zero, with lower values indicating better clustering. In this work, we use the inverse of the DB Index to make it consistent with other indices used in this research and to simplify result analysis and visualization.

A drawback of using internal criteria in cluster evaluation is that high scores on an internal measure do not necessarily indicate better performance, particularly for certain tasks like information

retrieval [37]. Additionally, this evaluation is biased towards algorithms that use the same cluster model. For example, k-means clustering naturally optimizes object distances, and a distance-based internal criterion will likely overrate the resulting clustering. We, therefore, also consider external measures in our analysis.

3.5.2 External Evaluation: The external evaluation measures gauge the degree to which cluster labels match class labels supplied externally [42]. We have two sets of class labels to consider; scam tags (Section 3.1.3) and phish/benign tags (Section 3.1.2). The former is used to evaluate the clustering performance with respect to scam identification and the latter with respect to phishing/benign classification. Note that the emails reported in real-time will not have any such tags. Hence, external measures are only useful during the development stage of the algorithm.

Adjusted Rand Index [14]: The Rand Index (RI) computes a similarity measure between two clustering outputs by considering all pairs of samples and counting pairs that are assigned in the same or different clusters in the predicted and true labels. The raw RI score is then "adjusted for chance" into the Adjusted Rand Index (ARI) score. ARI values range between 0.0 to 1.0, with higher values indicating more similarity between the two clustering methods. We use ARI with the scam tags as the class labels.

Adjusted Mutual Information [60]: The mutual information (MI) of two random variables is a measure of the mutual dependence between the two variables. It determines how different the joint distribution of the pair (X, Y) is from the product of the marginal distributions of X and Y. Adjusted Mutual Information (AMI) is an adjustment of the Mutual Information (MI) score to account for chance. AMI values range between 0.0 to 1.0, with higher values indicating more similarity between the two clustering methods. We use AMI with the scam tags as the class labels.

Validity Measure [48]: The V-measure is the harmonic mean between homogeneity and completeness of the clusters formed. A perfectly homogeneous clustering is one where each cluster has data-points belonging to the same class label and a perfectly complete clustering is one where all data-points belonging to the same class are clustered into the same cluster. We use validity to measure the homogeneity and completeness of the clusters with respect to the distribution of the scams.

Purity: Purity is ratio of the total number of data points belonging to the dominant class in a cluster to that of its size. Scores closer to 1.0 suggest better clustering accuracy. We use purity as a measure of homogeneity of phish/benign emails in each cluster. We calculate the ratio of the dominant class for each cluster and take an average over all the clusters.

4 EVALUATION

In this section, we provide our quantitative and qualitative findings as a result of our experiments. Our quantitative results are based on the external and internal measures described in previous sections. We further conduct a detailed post-clustering analysis to investigate the relevancy of the feature sets we are using.

4.1 Quantitative Evaluation

We experimented with the three clustering algorithms by using various features sets as explained in Table 3. We now share the results for each of the clustering algorithms, and we report the number of clusters formed and results of evaluation metrics on each feature set in the corresponding tables. The first column of each results table [4 5 6] is the name of the feature set and the number of clusters formed, the second, third and fourth column give the external evaluation measures, followed by the purity measure and the last two columns give the internal evaluation measures. The first row in each table corresponds to the results of the baseline feature set followed by the other feature sets.

K-Means Clustering: Table 4 shows that K-Means, in general, performs better than the other two clustering methods. It gives betters results than DBSCAN and Agglomerative clustering with four of the seven feature sets. With respect to the external evaluation metrics, K-Means works best with f_{es} comprising the BERT vectors for the email body text and the subject-line topic models, implying that with this combination of features, it can group email scams most efficiently. In terms of purity, K-Means has an average accuracy of 98%, which is the lowest of the three methods.

DBSCAN Clustering: DBSCAN performs worse than the other two clustering methods as shown in Table 5. With respect to the external evaluation metrics, DBSCAN works best with f_{eu} comprising the BERT vectors for the email body text and the URL features. It has the lowest external evaluation values of all, indicating a lower ability to identify phishing scams in a dataset. In terms of purity of phish/benign, DBSCAN shows a good performance with approximately 99.3% accuracy, which means it can distinguish between phish and benign emails most of the time. Although, a major drawback observed with this method is that it classifies more than half the data points as noise. This result is because the DBSCAN algorithm fails in cases of varying density clusters, due to the fixed cluster radius aspect. The Silhouette score values are also very low (close to zero), indicating poorly formed overlapping clusters. A surprising observation in DBSCAN clustering was high values for the Davies–Bouldin index, as opposed to the very low values of the Silhouette score. This result is a consequence of the fact that more than half the points are labelled noise and the remaining data points form well-separated clusters.

Agglomerative Clustering: In Table 6, we see that Agglomerative clustering performs better than DBSCAN and similar to K-Means. It achieves good results with f_{es}, f_{esu} and f_{esh} while outperforming K-Means on some sets. With respect to the external evaluation metrics, it shows the best accuracy with f_{es} comprising the BERT vectors for the email body text and the subject-line topic models, with the highest values of all the clustering combinations. The average phish/benign purity for this method is 98.7%, which indicates relatively better ability to identify scams. The Silhouette Score and DB-Index are also better for this method than the other two on the seven feature sets, indicating better cluster formation.

4.1.1 Results of Feature Selection. We use Chi-Square feature selection (Section 3.4) on all the features extracted to measure the correlation of each feature with the email's phish/benign tag and scam tag. We run the three clustering algorithms with the identified Chi-Square feature subset. The results of this clustering are lower

than the original set, and hence we disregard the feature selection results and proceed with the original feature sets.

4.2 Qualitative Analysis

To better understand the effectiveness of each clustering algorithm, we conduct a detailed post-clustering analysis of the most accurate feature set for each. We analyse the cluster sizes and the phish/benign homogeneity of each cluster. We identify the cluster with the least phish/benign purity score and understand the spread of emails in the cluster. We further analyse the distribution and separation of the scam tags, to gauge the efficacy of the algorithm to identify scams.

K-Means Clustering: K-Means clustering performs best with f_{es} and we further analyse the resulting clusters. We identify 70 as the optimal number of clusters for this combination from the parameter sweep (Figure 2b). The cluster size varies from 8 to 240 with an average of 62 emails per cluster. With respect to the homogeneity of phish/benign emails, 54 of the 70 clusters are either fully phish or fully benign. The worst performing cluster contains 70 emails with 31.4% phishing emails and 68.6% benign emails. The phishing emails in this cluster are all emails with random text, some that appear to be excerpts from a book, and lack the usual phishing email structure and vocabulary. With respect to scam detection, 7 of the 14 tags outlined in Table 1 have been grouped fully into distinct clusters. The worst tag distribution is for Tag-5 ('quickly restore full access') which is spread over 6 clusters. On further analysis, we observe that most of the emails separated into different clusters have different organization names.

DBSCAN Clustering: DBSCAN clustering performs best with f_{eu} and we further analyse the resulting clusters. We use 8 as the minimum number of samples and identify 9.4 as the optimal epsilon value using the parameter sweep (Figure 2c), resulting in 50 clusters. The cluster size varies from 8 to 309 with an average of 25 emails per cluster. With respect to the homogeneity of phish/benign emails, all the clusters formed are either fully phish or fully benign. Although the clusters formed are perfectly phish or benign, around 71% of the emails were labelled noise, creating a major drawback for this method. This includes 52% of the phishing dataset, most of which are obvious phishing attacks. With respect to scam detection, 6 of the 14 tags outlined in Table 1 have been grouped fully into distinct clusters. Further, 8 of the 14 tags had datapoints labelled as noise. The worst tag distribution was for Tag-5 ('quickly restore full access') which is spread over 4 clusters and noise. The emails spread in different clusters have same organization name and email structure, indicating poor performance for DBSCAN.

Agglomerative Clustering: Agglomerative clustering performs best with f_{es} and we further analyse the resulting clusters. We identify 44 as the optimal number of clusters for this combination from the parameter sweep (Figure 2a). The cluster size ranges from 12 to 309 with an average of 99 emails per cluster. With respect to the homogeneity of phish/benign emails, 33 of the 44 clusters are either fully phish or fully benign. The worst performing cluster contains 117 emails with 80.34% benign emails and 19.66% phishing emails. The phishing emails in this cluster mostly contain junk text (e.g., non-meaningful text), random words and short phrases that seem to be excerpts from articles or books. It also comprises a few emails that are obvious phishing attempts but are not structured as the majority of phishing emails. With respect to scam detection, 7 of the 14 tags outlined in Table 1 have been grouped fully into distinct clusters. The worst tag distribution is for Tag-13 ('safety and integrity') which is spread over 4 clusters. Similar to K-Means clustering, the emails placed in different clusters have a similar email scam and outline but impersonate a different organization.

4.3 Summary

The quantitative analysis shows promising results for using unsupervised clustering to group phishing emails by the underlying scams. With respect to overall performance, we see that K-Means performs better with most feature sets. We also identify *Agglomerative Clustering* + f_{es} as the best combination to identify phishing scams in a dataset. We observe that feature sets comprising of both BERT vectors for the email body text and the subject-line topic models performed better, especially for K-Means and Agglomerative clustering. Both these features are context-oriented and support our approach of using context-based features. This is further reinforced by the results which show that the baseline consisting of only header and URL features, without any context information, performs much worse than those with contextual information. Another important observation we make is that the internal evaluation scores, in general, were lower for all three clustering methods indicating poorly-formed overlapping clusters. This can be attributed to the nature of our data and a generally high overlap in phishing email structures and text. This result is also expected since sophisticated AI filters do struggle in classifying phish, as they may look very similar to legitimate emails.

Phish/benign homogeneity. The clusters resulting from the three clustering algorithms show good performance with an overall average purity of 98.65%. Most of the phishing emails grouped in benign clusters are observed to have short text, random words and phrases that seem to be excerpts from articles or books. The text in these emails is more similar to the Enron emails, which correspond to day-to-day interactions between employees, that the usual structure and vocabulary of phishing emails.

Scam tags. For the distribution of scam tags, all three algorithms were able to identify and fully separate at least 6 of the 14 tags. We observe that tags 5 and 13 were particularly hard for the algorithms to cluster. This can the attributed to a higher variation in the emails of these scams, like different organization names and vocabulary.

5 CONCLUSION

This paper attempts to explore the feasibility of grouping reported phishing emails, based on certain similarities, to help organizations and IS teams efficiently mitigate phishing attacks. We conduct a comparative study on different combinations of clustering methods and feature sets for the identification and grouping of phishing scams from an email dataset. We extract a combination of contextual and semantic features from emails and perform three clustering algorithms (K-Means, DBSCAN, and Agglomerative) with seven distinct feature sets. Using a range of internal and external validation methods, we identify that K-Means clustering performs relatively better than the other two. We identify the best clustering combination to be Agglomerative Clustering with Feature Set

Table 4: K-Means Cluster Comparison

	External Measures				Internal Measures	
Feature Set	**ARI**	**AMI**	**Validity**	**Purity**	**Silhouette**	**DBI^{-1}**
f_b (56)	0.190	0.441	0.535	0.983	0.631	0.983
f_e (64)	0.851	0.817	0.836	0.975	0.105	0.352
f_{eh} (50)	0.805	0.788	0.806	0.980	0.120	0.332
f_{eu} (60)	0.818	0.803	0.822	0.972	0.096	0.353
f_{es} (70)	0.853	**0.825**	**0.843**	0.985	0.098	0.365
f_{esu} (56)	0.826	0.779	0.798	0.981	0.098	0.355
f_{esh} (49)	**0.855**	0.811	0.828	0.986	0.090	0.369
f_{eshu} (35)	0.817	0.780	0.795	0.980	0.076	0.340

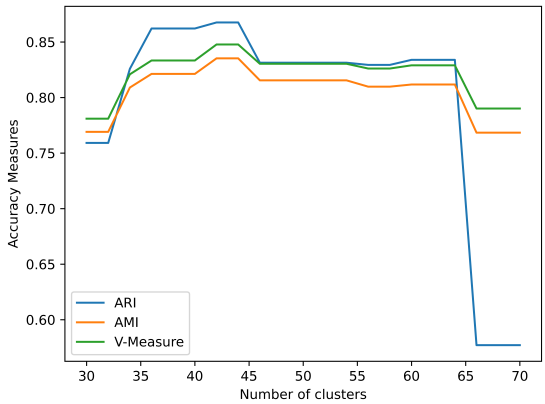
Table 5: DBSCAN Cluster Comparison

	External Measures				Internal Measures	
Feature Set	**ARI**	**AMI**	**Validity**	**Purity**	**Silhouette**	**DBI^{-1}**
f_b (28)	0.174	0.358	0.411	0.982	0.365	0.712
f_e (45)	0.649	0.745	0.768	0.993	0.032	0.743
f_{eh} (45)	0.648	0.744	0.768	0.992	0.017	0.725
f_{eu} (51)	**0.652**	**0.770**	**0.793**	0.993	0.003	0.783
f_{es} (45)	0.643	0.740	0.763	0.992	0.014	0.719
f_{esu} (47)	0.641	0.745	0.767	0.992	0.010	0.716
f_{esh} (46)	0.634	0.738	0.762	0.992	0.007	0.709
f_{eshu} (46)	0.629	0.767	0.789	0.992	-0.021	0.736

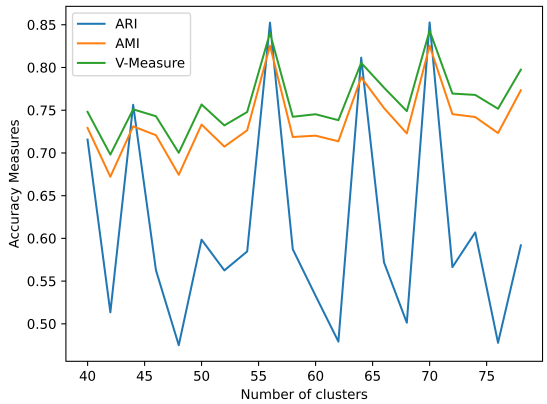
Table 6: Agglomerative Cluster Comparison

	External Measures				Internal Measures	
Feature Set	**ARI**	**AMI**	**Validity**	**Purity**	**Silhouette**	**DBI^{-1}**
f_b (48)	0.206	0.447	0.530	0.980	0.631	0.967
f_e (54)	0.747	0.780	0.798	0.987	0.133	0.388
f_{eh} (59)	0.735	0.784	0.801	0.987	0.132	0.384
f_{eu} (52)	0.752	0.783	0.801	0.987	0.127	0.374
f_{es} (44)	**0.867**	**0.835**	**0.848**	0.987	0.104	0.351
f_{esu} (44)	0.862	0.823	0.838	0.988	0.103	0.349
f_{esh} (44)	0.835	0.820	0.833	0.988	0.106	0.363
f_{eshu} (34)	0.760	0.786	0.800	0.987	0.091	0.333

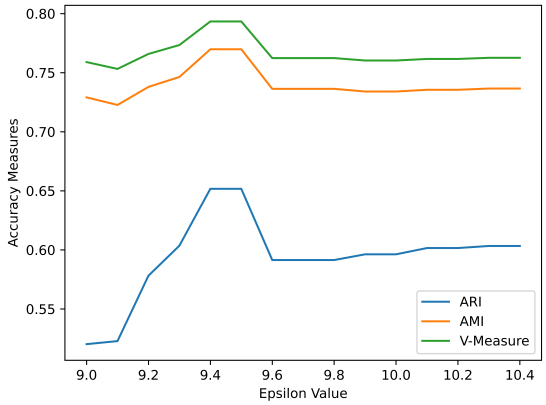
4 comprising the BERT vectors for the email body text and the subject-line topic models. Our paper shows that by grouping together emails of similar scams we can significantly reduce the load on the IS staff, who can now respond to a fewer number of clusters rather than each individual email. We also show that unsupervised clustering is a promising approach for such scam grouping. Such a clustering model can greatly help IS staff and support teams better manage phishing reports, and help users get relevant and useful advice in a timely manner.



(a) Agglomerative Clustering



(b) K-Means Clustering



(c) DBSCAN Clustering

Figure 2: Results of the Hyper-Parameter optimizations. (a) Parameter sweep for Agglomerative Clustering for the number of clusters. (b) Parameter sweep for K-Means Clustering for the number of clusters. (c) Parameter sweep for DBSCAN Clustering for the epsilon value.

Future directions for this work include adapting the algorithms to newer datasets. Phishing scams have also evolved resulting in new scam types. Conducting more experiments on newer datasets and reported phishing datasets can give very useful insights. The experimental results show that for both K-Means and Agglomerative clustering, adding header and URL features to the feature sets causes a drop in the accuracy score. This result is surprising because, in general, both URL and header features are considered important features in phishing research and have been widely used in various tasks. In our future work, we want to find more efficient ways to represent and utilize these features. Another important observation made is that most phishing emails clustered with benign emails were short text emails, some with random words and junk text. An important future research direction is to define a semantic or coherence score to identify such scams. Further, the results indicate that the algorithms used provide a low Silhouette and DB score, which usually indicates overlapping within the clusters. Thus there is a possibility that we have closely-related scams separated into different clusters, such as those with different organization names. In our future work, we plan on using alternate techniques like soft-clustering algorithms to address this issue.

ACKNOWLEDGMENTS

This work was funded in part by the UKRI Strategic Priorities Fund via the REPHRAIN Research Centre. The authors would also like to thank the TULiPS research group for the continual feedback.

REFERENCES

[1] Mamoun Alazab, Robert Layton, Roderic Broadhurst, and Brigitte Bouhours. 2013. Malicious spam emails developments and authorship attribution. In *2013 fourth cybercrime and trustworthy computing workshop*. IEEE, 58–68.

[2] Izzat Alsmadi and Ikdam Alhami. 2015. Clustering and classification of email contents. *Journal of King Saud University - Computer and Information Sciences* 27, 1 (2015), 46–57. https://doi.org/10.1016/j.jksuci.2014.03.014

[3] Kholoud Althobaiti, Adam D G Jenkins, and Kami Vaniea. 2021. A Case Study of Phishing Incident Response in an Educational Organization. *Proc. ACM Hum.-Comput. Interact.* 5, CSCW2, Article 338 (oct 2021), 32 pages. https://doi.org/10.1145/3476079

[4] Kholoud Althobaiti, Ghaidaa Rummani, and Kami Vaniea. 2019. A review of human-and computer-facing url phishing features. In *2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 182–191.

[5] Mihael Ankerst, Markus M Breunig, Hans-Peter Kriegel, and Jörg Sander. 1999. OPTICS: Ordering points to identify the clustering structure. *ACM Sigmod record* 28, 2 (1999), 49–60.

[6] APWG. 2022. *Phishing Activity Trends Report, 1st Quarter 2022*. Retrieved June 17, 2022 from https://docs.apwg.org//reports/apwg_trends_report_q1_2022.pdf

[7] Eint Sandi Aung, Chaw Thet Zan, and Hayato Yamana. 2019. A survey of URL-based phishing detection. In *DEIM Forum*. G2–3.

[8] Abdul Basit, Maham Zafar, Xuan Liu, Abdul Rehman Javed, Zunera Jalil, and Kashif Kifayat. 2021. A comprehensive survey of AI-enabled phishing attacks detection techniques. *Telecommunication Systems* 76, 1 (2021), 139–154.

[9] Simon Bell and Peter Komisarczuk. 2020. An analysis of phishing blacklists: Google safe browsing, openphish, and phishtank. In *Proceedings of the Australasian Computer Science Week Multiconference*. 1–11.

[10] Eduardo Benavides, Walter Fuertes, Sandra Sanchez, and Manuel Sanchez. 2020. Classification of phishing attack solutions by employing deep learning techniques: A systematic literature review. *Developments and advances in defense and security* (2020), 51–64.

[11] Panagiotis Bountakas, Konstantinos Koutroumpouchos, and Christos Xenakis. 2021. A Comparison of Natural Language Processing and Machine Learning Methods for Phishing Email Detection. In *The 16th International Conference on Availability, Reliability and Security*. 1–12.

[12] William W. Cohen. 2000. Enron Email Dataset. https://www.cs.cmu.edu/~enron/

[13] David L Davies and Donald W Bouldin. 1979. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence* 2 (1979), 224–227.

[14] Rogério Rodrigues de de Vargas and Benjamín Rene Callejas Bedregal. 2013. A way to obtain the quality of a partition by adjusted Rand index. In *2013 2nd Workshop-School on Theoretical Computer Science*. IEEE, 67–71.

[15] Dave DeBarr, Venkatesh Ramanathan, and Harry Wechsler. 2013. Phishing detection using traffic behavior, spectral clustering, and random forests. In *2013 IEEE International Conference on Intelligence and Security Informatics*. 67–72. https://doi.org/10.1109/ISI.2013.6578788

[16] L Jegatha Deborah, R Baskaran, and A Kannan. 2010. A survey on internal validity measure for cluster validation. *International Journal of Computer Science & Engineering Survey* 1, 2 (2010), 85–102.

[17] Department for Digital, Culture, Media and Sport. 2021. Cyber Security Breaches Survey 2021. https://bit.ly/3IaqF6t

[18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[19] Son Dinh, Taher Azeb, Francis Fortin, Djedjiga Mouheb, and Mourad Debbabi. 2015. Spam campaign detection, analysis, and investigation. *Digital Investigation* 12 (2015), S12–S21.

[20] Ayman El Aassal, Shahryar Baki, Avisha Das, and Rakesh M Verma. 2020. An in-depth benchmarking and evaluation of phishing detection research for security needs. *IEEE Access* 8 (2020), 22170–22192.

[21] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (Portland, Oregon) *(KDD'96)*. AAAI Press, 226–231.

[22] Yong Fang, Cheng Zhang, Cheng Huang, Liang Liu, and Yue Yang. 2019. Phishing email detection using improved RCNN model with multilevel vectors and attention mechanism. *IEEE Access* 7 (2019), 56329–56340.

[23] Ian Fette, Norman Sadeh, and Anthony Tomasic. 2007. Learning to detect phishing emails. In *Proceedings of the 16th international conference on World Wide Web*. 649–656.

[24] D Asir Antony Gnana, S Appavu Alias Balamurugan, and E Jebamalar Leavline. 2016. Literature review on feature selection methods for high-dimensional data. *International Journal of Computer Applications* 136, 1 (2016), 9–17.

[25] Eder S Gualberto, Rafael T De Sousa, P De B Thiago, João Paulo CL Da Costa, and Cláudio G Duque. 2020. From feature engineering and topics models to enhanced prediction rates in phishing detection. *Ieee Access* 8 (2020), 76368–76385.

[26] Peter Haider and Tobias Scheffer. 2009. Bayesian clustering for email campaign detection. In *Proceedings of the 26th Annual International Conference on Machine Learning*. 385–392.

[27] Isredza Rahmi A. Hamid and Jemal H. Abawajy. 2013. Profiling Phishing Email Based on Clustering Approach. In *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*. 628–635. https://doi.org/10.1109/TrustCom.2013.76

[28] Isredza Rahmi A. Hamid and Jemal H. Abawajy. 2014. An Approach for Profiling Phishing Activities. *Comput. Secur.* 45 (sep 2014), 27–41. https://doi.org/10.1016/j.cose.2014.04.002

[29] Martin Husak and Jakub Cegan. 2014. PhiGARo: Automatic Phishing Detection and Incident Response Framework. In *2014 Ninth International Conference on Availability, Reliability and Security*. IEEE. https://doi.org/10.1109/ares.2014.46

[30] IRONSCALES Ian Thomas. 2021. IRONSCALES: The State of Cybersecurity Survey. Retrieved February 28th, 2022 from https://ironscales.com/blog/ironscales-releases-findings-from-state-of-cybersecurity-survey/

[31] Kalpana D. Joshi and Prakash S. Nalwade. 2013. Modified K-Means for Better Initial Cluster Centres.

[32] Asif Karim, Sami Azam, Bharanidharan Shanmugam, and Krishnan Kannoorpatti. 2020. Efficient Clustering of Emails Into Spam and Ham: The Foundational Study of a Comprehensive Unsupervised Framework. *IEEE Access* (2020), 154759–154788. https://doi.org/10.1109/ACCESS.2020.3017082

[33] Mahmoud Khonji, Youssef Iraqi, and Andrew Jones. 2013. Phishing detection: a literature survey. *IEEE Communications Surveys & Tutorials* 15, 4 (2013), 2091–2121.

[34] Erka Koivunen. 2010. "Why Wasn't I Notified?": Information Security Incident Reporting Demystified. 55–70. https://doi.org/10.1007/978-3-642-27937-9_5

[35] Christian Ludl, Sean McAllister, Engin Kirda, and Christopher Kruegel. 2007. On the effectiveness of techniques to detect phishing sites. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*. Springer, 20–39.

[36] J MacQueen. 1967. Classification and analysis of multivariate observations. In *5th Berkeley Symp. Math. Statist. Probability*. 281–297.

[37] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press. https://doi.org/10.1017/cbo9780511809071

[38] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[39] J Nazario. 2005. Nazario Phishing Corpus. https://monkey.org/~jose/phishing/

[40] Frank Nielsen. 2016. Hierarchical clustering. In *Introduction to HPC with MPI for Data Science*. Springer, 195–211.

[41] Diegoocampoh Ocampo. 2017. GitHub - diegoocampoh. https://github.com/diegoocampoh/MachineLearningPhishing

[42] Julio-Omar Palacio-Niño and Fernando Berzal. 2019. Evaluation metrics for unsupervised learning algorithms. *arXiv preprint arXiv:1905.05667* (2019).

[43] Danny Palmer. 2019. These are the 12 most common phishing email subject lines cyber criminals use to fool you. https://www.zdnet.com/article/these-are-the-12-most-common-phishing-email-subject-lines-cyber-criminals-use-to-fool-you/

[44] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. 2018. Tranco: A research-oriented top sites ranking hardened against manipulation. *arXiv preprint arXiv:1806.01156* (2018).

[45] Anand Rajaraman and Jeffrey David Ullman. 2011. *Data Mining*. Cambridge University Press, 1–17. https://doi.org/10.1017/CBO9781139058452.002

[46] Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, Vol. 242. New Jersey, USA, 29–48.

[47] Lior Rokach and Oded Maimon. 2005. Clustering methods. In *Data mining and knowledge discovery handbook*. Springer, 321–352.

[48] Andrew Rosenberg and Julia Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*. 410–420.

[49] Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics* 20 (1987), 53–65.

[50] Said Salloum, Tarek Gaber, Sunil Vadera, and Khaled Shaalan. 2021. Phishing email detection using natural language processing techniques: a literature survey. *Procedia Computer Science* 189 (2021), 19–28.

[51] Erich Schubert, Jörg Sander, Martin Ester, Hans Peter Kriegel, and Xiaowei Xu. 2017. DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN. *ACM Trans. Database Syst.* 42, 3, Article 19 (jul 2017), 21 pages. https://doi.org/10.1145/3068335

[52] IBM Security. 2021. Cost of a Data Breach Report 2021. Retrieved June 30, 2022 from https://www.ibm.com/security/data-breach

[53] Aakanksha Sharaff and Naresh Kumar Nagwani. 2020. ML-EC2. *International Journal of Web-Based Learning and Teaching Technologies* 15, 2 (April 2020), 19–33. https://doi.org/10.4018/ijwltt.2020040102

[54] Jianbo Shi and Jitendra Malik. 2000. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence* 22, 8 (2000), 888–905.

[55] Suzanna Sia, Ayush Dalmia, and Sabrina J Mielke. 2020. Tired of topic models? clusters of pretrained word embeddings make for fast and good topics too! *arXiv preprint arXiv:2004.14914* (2020).

[56] M Somesha and Alwyn R Pais. 2022. Classification of Phishing Email Using Word Embedding and Machine Learning Techniques. *Journal of Cyber Security and Mobility* (2022), 279–320.

[57] Amber Van Der Heijden and Luca Allodi. 2019. Cognitive triaging of phishing attacks. In *28th USENIX Security Symposium (USENIX Security 19)*. 1309–1326.
[58] Verizon. 2021. Verizon 2021 Data Breach Investigations Report. https://www.verizon.com/business/resources/reports/2021/2021-data-breach-investigations-report.pdf
[59] Rakesh Verma, Narasimha Shashidhar, and Nabil Hossain. 2012. Detecting phishing emails the natural language way. In *European Symposium on Research in Computer Security*. Springer, 824–841.
[60] Nguyen Xuan Vinh, Julien Epps, and James Bailey. 2010. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *The Journal of Machine Learning Research* 11 (2010), 2837–2854.
[61] Chun Wei, Alan Sprague, Gary Warner, and Anthony Skjellum. 2008. Mining spam email to identify common origins for forensic application. In *Proceedings of the 2008 ACM symposium on Applied computing*. 1433–1437.
[62] Carsten Willems, Thorsten Holz, and Felix Freiling. 2007. Toward automated dynamic malware analysis using cwsandbox. *IEEE Security & Privacy* 5, 2 (2007), 32–39.
[63] Harry Williams, Orla Leggett, Nick Coleman, Jayesh Navin Shah, and Steven Furnell. 2021. *Cyber Security Breaches Survey 2021 – Chapter 5: Incidence and impact of breaches or attacks.* Technical Report. National Cyber Security Centre.
[64] Yang Xiang. 2008. Managing email overload with an automatic nonparametric clustering system. *The Journal of Supercomputing* 48, 3 (July 2008), 227–242. https://doi.org/10.1007/s11227-008-0216-y
[65] Masoumeh Zareapoor and KR Seeja. 2015. Feature extraction or feature selection for text classification: A case study on phishing email detection. *International Journal of Information Engineering and Electronic Business* 7, 2 (2015), 60.
[66] Kim Zetter. 2016. Inside the Cunning, Unprecedented Hack of Ukraine's Power Grid. Retrieved July 15, 2022 from https://www.wired.com/2016/03/inside-cunning-unprecedented-hack-ukraines-power-grid/
[67] Yujia Zhai, Wei Song, Xianjun Liu, Lizhen Liu, and Xinlei Zhao. 2018. A chi-square statistics based feature selection method in text classification. In *2018 IEEE 9th International conference on software engineering and service science (ICSESS)*. IEEE, 160–163.
[68] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. 1996. BIRCH: an efficient data clustering method for very large databases. *ACM sigmod record* 25, 2 (1996), 103–114.

A SUBJECT TOPIC MODEL

To create context-based representation vectors for the subject line, we first generate a topic model from the subject lines and then use the keywords corresponding to each topic to generate topic vectors. We propose two algorithms, and both use the same set of parameters as an input: S_p, S, the set of preprocessed subject lines (Section 3.1.2); n, the number of topics to be generated; and x, the number of top keywords to be extracted from a topic. Algorithm 1 provides the pseudocode for topic model generation. This algorithms uses the phishing dataset subject-lines (S_p) as input and the resulting output would be the set TM of n-topics, where each element is the set of the top x keywords for a topic. Using this algorithm, we generated 30 topics and 20 keywords per topic (i.e., n=30 and x=20). Algorithm 2 provides the pseudocode for generating topic vectors for all the subject lines (S) from the model generated in Algorithm 1. Here we use a bag-of-topics approach, where we count the number of keywords in each topic that occur in the subject line.

Algorithm 1: genSLTopics(S_p, n, x)

```
Output: TM, set of topics with corresponding keywords
T ← {}, TM ← []
foreach s ∈ S_p do
    T_s ← gen3Grams(s)
    T ← T ∪ T_s
T_bert ← getBertEmb(T)
C_n ← applyKMeans(n, T_bert)
foreach c ∈ C_n do
    K_c ← getTopXkeywords(c, x)
    TM ← TM.append(K_c)
return TM
```

Algorithm 2: genSLVectors(S, n, x)

```
Output: V, set of subject line vectors
V ← []
TM ← genSLTopics(S, n, x)
for s ∈ S do
    V_s ← []
    for T ∈ TM do
        k ← 0
        foreach kw ∈ T do
            if kw ∈ s then
                k ← k + 1
        V_s ← V_s.append(k)
    V ← V.append(V_s)
return V
```