Applied Information Technology And Computer Science Vol. 4 No. 2 (2023) 248-265

AITCS

Homepage: http://publisher.uthm.edu.my/periodicals/index.php/aitcs
e-ISSN :2773-5141

Feature Extraction Tool for Phishing Dataset

Kee Xue Nie[1], Isredza Rahmi A Hamid[1]*

[1]Fakulti Sains Komputer dan Teknologi Maklumat,
Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, 86400, MALAYSIA

DOI: https://doi.org/10.30880/aitcs.2023.04.02.015
Received 01 November 2023; Accepted 07 November 2023; Available online 30 November 2023

Abstract: Feature extraction is a fundamental technique applied by researchers to extract the dimensionality of dataset. Machine learning algorithms rely on feature vectors rather than raw data for effective processing. Thus, the development of feature extraction tools become crucial in converting raw text datasets into feature vectors. In this project, we propose the utilization of Word2vec model and object-oriented approach to extract features from raw text datasets. Our tool could be applied in research exploring text-based feature extraction, with the implementation realized through Python. The feature extraction tool offers the flexibility to extract user-selected features or all phishing-related features from the raw text dataset, enhancing its applicability in diverse research scenarios. Features that are included are Contains_Urgency, Contains_Free, Contains_Exclamation_Marks, Contains_Urls, Word_Count, Contain_Most_Similar_Words_To_Urgent_*Word*, and Contain_Most_Frequent_Word_*Word*. Features such as Contain_Most_Similar_Words_To_Urgent_*Word* and Contain_Most_Frequent_Word_*Word* are extracted using the Word2vec model. Main modules that are included in the tool are the user register module, user login module, upload module, download module, and feature extraction module.

Keywords: Feature Extraction, Word2vec, Phishing

1. Introduction

Phishing is a social engineering crime where users are targeted to disclose their sensitive personal information through fraudulent messages. Attackers employ various tactics, such as sending deceptive emails or text messages that appear to originate from trusted sources, in order to manipulate victims. According to the Phishing Activity Trends Report [1], the incidence of phishing attacks increased a linear upward trend from the fourth quarter of 2021 to the third quarter of 2022.

Machine learning, an artificial intelligence technique, is commonly used by researchers to solve tasks [2]. In the context of phishing detection, machine learning algorithms rely on extracted features from raw data for training and testing purposes [3]. These algorithms analyse the characteristics of words and assess the accuracy of detecting phishing content within text. Feature extraction plays a crucial role in reducing the time required to convert raw data into feature vectors.

*Corresponding author: rahmi@uthm.edu.my

Feature extraction techniques involve reducing the dimensionality of a feature space, replacing the existing features with a new set of reduced features [3]. Popular methods for feature extraction are Principal Components Analysis (PCA), Latent Semantic Analysis (LSA), Bag-of-Words (BoW), Term Frequency-Inverse Document Frequency (TF-IDF), and the Word2vec model, which combines the Continuous Bag of Word (CBOW) and Skip-gram models.

While feature extraction methods such as BoW and TF-IDF are popular among researchers [4], [5] They possess certain limitations. Firstly, these methods cannot effectively extract semantic meaning features, which are crucial for accurately classifying and detecting phishing text. Secondly, open compound words are extracted as separate entities, resulting in a loss of the original compound word's intended meaning, and adversely impacting the extracted features. Besides that, existing online text-based feature extraction tools such as YAKE! Keyword Extractor [6], can only extract keywords according to predefined features. To address these limitations, the Word2vec model is chosen for the development of feature extraction tool. The Word2vec model allows vectors to carry semantic analogies according to the cosine distance measurements [7]. This model uses vectors to effectively preserves the meaning of compound words [8]. The objectives of this project are as follow:

i. to design a feature extraction tool to extract phishing dataset,
ii. to develop a feature extraction tool by implementing the Word2vec model, and
iii. to test the feature extraction tool with phishing dataset in terms of user acceptance and system functionality.

The feature extraction tool will be developed using Python and adhere to the principles of object-oriented system development (OOSD) methodology. The intended users for this tool are researcher engaged in phishing-related works. The tool allows users to upload text file or insert text for feature extraction using the Word2vec model, and subsequently download the extracted features in a separate file. Users will be able to obtain a new dataset comprising word vectors derived from original raw dataset. The feature extraction will greatly assist researchers engaged in text-based research by facilitating precise extraction of features related to phishing topics.

The rest of the paper is organized as follows: Section 2 provides the literature review about phishing, Natural Language Processing (NLP) related to feature extraction, feature extraction techniques such as One Hot Encoding, BoW, TF-IDF, and Word2vec model. Existing feature extraction tools will be explained and analysed in Section 2 as well. Section 3 presents the methodology employed in this project. Section 4 explains the analysis and design of feature extraction tool. Section 5 encompasses the testing results and concludes the paper.

2. Related Work

Section 2 discusses the NLP, feature extraction and its technique, and the existing text-based feature extraction tools.

2.1 Natural Language Processing (NLP)

NLP refer to the ability of computers to understand human language [9]. In the context of text-based feature extraction, NLP plays a crucial role in converting raw text into word vectors that computers can understand. NLP tasks shows significant assistance in extracting feature from text. The first important task in NLP is word sense disambiguation. The task involves determining the most appropriate meaning for a word, helping to identify the most suitable interpretation. Another significant task is the sentiment analysis, which involves extracting characteristics like emotion, sarcasm, and attitude from text. Machine learning algorithms can scan text message to indicate whether if they are phishing or not. Indicators for phishing text may include grammar errors, threatening language, inappropriate urgency, or others. These indicators can be found in the features extracted from the raw data.

2.2 Feature Extraction

Feature extraction is the preliminary process of Machine Learning technique [4]. Feature extraction is crucial to comprehending the context of given data [5]. The output of feature extraction is the vector space model in numeric data, which could be used to train and test machine learning algorithms. Feature extraction could be done either in manual or automatically. Manual feature extraction requires users' understanding in the background or domain of the related topic, whereas automated feature extraction applies specific techniques to get outcome quickly.

2.3 Feature Extraction Technique

Feature extraction technique provides a new vector space model once the raw data is pre-processed by applying tokenization, stop-word removal, stemming, and lemmatization [10]. After pre-processing, the data can be passed into the feature extraction method. The CBOW architecture enables computers to understand the context of the words by converting them into vectors and passing them to the hidden layer. This process generates predicted outputs with accurate context [11]. On the other hand, the skip-gram model predicts the context word given a specific word, which is the opposite of CBOW. The target word is passed into the hidden layer to perform multiplication with the weight matrix. This result is then multiplied with the weight matrix in the output layer, followed by the application of the softmax activation function to calculate the probability of words appearing in the target word. For this project, the word2vec model is implemented. This model facilitates precise analogical reasoning and enhances the quality of word representations [8]. It combines the CBOW and skip-gram model.

2.4 Existing Feature Extraction Tools

Various feature extraction tools available such as Linguistic Feature Extraction Tool [12], PhisherCop [13] and Phishing Email Detection Tool [3]. The Linguistic Feature Extraction Tool [12] applied BoW and LSA, Decision Tree, and Sentiment Analysis techniques for feature extraction. BoW provides word frequency; LSA reveals the contextual meaning of the words; Decision Tree addresses regression and classification issues; Sentiment Analysis classifies text based on emotions polarity. The selected features for extraction include the Use of the Pronouns, Emoticons, Mentions, Contains Intensifier, Contains Spread, Contains Hashtags, Informativeness, Contains Slang, Contains URLs, and Flesch Reading Ease Readability Score. Users can input the raw text into the provided textbox, selected desired features for extraction, and choose between English and Arabic languages. The results are displayed in the user interface. However, the disadvantage of this tool is the inability to download the features extracted.

PhisherCop [13] applied TF-IDF technique for feature extraction. The TF-IDF vector is compared to identify similarities between text messages using cosine similarity. This helps efficiently summarize the text by determining the most relevant words to the overall topic. Hence, it extracted the most representative words as features for classification by normalizing the word frequency. The results are presented in a pie chart. PhisherCop allows users to input content, subject the sender's email address. Then, displays the phishing likelihood in a pie chart. PhisherCop collected Email Spam and Ham Corpus by Spam Assassin in Kaggle as their dataset [13].

The Phishing Email Detection tool from [3] used PCA, LSA and TF-IDF technique. TF-IDF showcases word frequency, PCA reduces data dimensionality, and LSA classifies words based on meaning. The original features are transformed into a smaller set through PCA, with eigenvectors extracted as new features. LSA identifies conceptual words by analysing relationships between terms that occur in similar contexts. This tool extracts body-based features, Uniform Resource Locator (URL) based features, and header-based features. The extraction output is a long vector.

2.5 Comparison Existing Tools with The Proposed Tool

Table 1 presents a comparison between existing feature extraction tools and out proposed tool. The PhisherCop tool [13] and the Phishing Email Detection tool described in [3] are similar in terms of the features extraction methods. However, the Linguistic Feature Extraction Tool [12] differs from the other tools as it does not employ a similar algorithm.

Our proposed tool has similar text pre-processing steps as PhisherCop and the Phishing Email Detection tool, which include tokenization and stop word removal. Both PhisherCop and Phishing Email Detection tool apply the TF-IDF method, which only extract the word frequency. In contrast, our proposed feature extraction tool utilizes the Word2vec model, which extracts word frequency and sentiment features specific to phishing datasets. By combining the feature extracted from the Linguistic Feature Extraction Tool and the Phishing Email Detection tool, our proposed encompasses word frequency, relationships between term and concepts, and emotions. These features provide researchers with meaningful insights to enhance detect phishing.

Table 1: The Comparison of Existing Tools with Proposed Tool

Work	Algorithms	Datasets	Features	Methods	Results
Linguistic Feature Extraction Tool [12]	Bag-of-Words Model, Latent Semantic Analysis, Decision Tree, Sentiment Analysis	Data from Twitter	Word Frequency, Lexicological Correlation, Total Number of Words in The Text, Emotions	**1.** Text preprocessing **2.** Feature Extraction Model **3.** Feature Extractor **4.** Linguistic Features	62.60% (English), 72.55% (Arabic)
PhisherCop [13]	TF-IDF	3,052 emails and 5,574 SMS from Kaggle	Word Frequency	1. Stop Word Removal 2. Tokenization 3. TF-IDF	Not Mentioned
Phishing Email Detection tool [3]	PCA, LSA, TF-IDF	2,700 emails from Monkey and Spam Assassin	Relationships between a term and concepts, Word Frequency	1. Tokenization 2. Stop Word Removal 3. Stemming 4. Term-Document-Frequency (TDF)	96% (PCA), 93.5% (LSA)
Proposed Tool	Word2vec	Data from Kaggle and Monkey	Relationships between a term and concepts, Word Frequency, Emotions	**1.** Stop Word Removal **2.** Tokenization **3.** Word2vec	

3. Methodology

For this project, we have chosen to use the Python programming language. Python is an Object-Oriented Programming Language (OOP), making the OOSD methodology an appropriate choice. OOSD is an approach to system development that places objects at the core of the project design. Figure 1 shows the four phases of OOSD: planning, analysis and design, implementation, and maintenance.

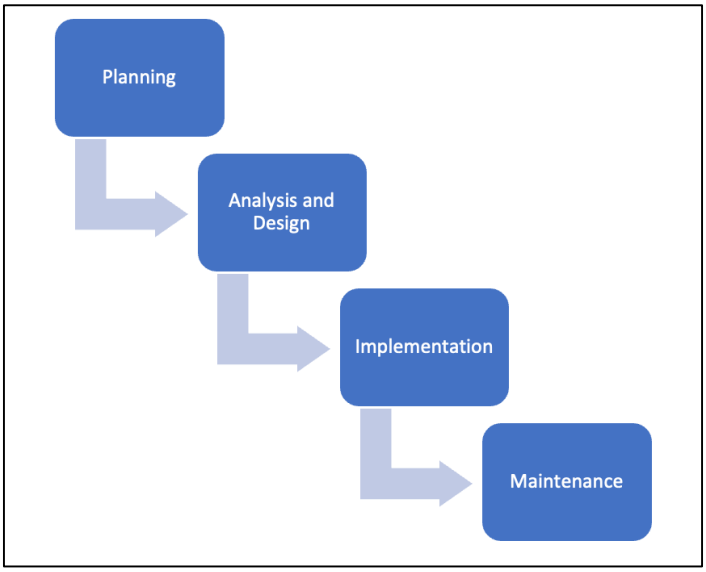

Figure 1: Phases in OOSD [14]

3.1 Planning Phase

The planning phase is the initial phase of the project, where all requirements and problems are defined. During this phase, problem statements are formulated through surveys and making comparison with the existing tool, aiming to address these problems through the current project. Objectives and project scope are established after clearly stating the problems to ensure that the project is carried out with the intended goals in mind. Additionally, the planning phase involves conducting a literature review, which entails searching and analysing relevant journal articles, conference papers, and books associated with the project. The deliverables of this phase include project, problem statements, objectives, project scope, and expected outcome of the project, which are stated in section 1.

3.2 Analysis and Design Phase

In the analysis and design phase, the system design is developed based on the outputs obtained from the Planning phase. The Graphical User Interface (GUI) of the Feature extraction tool using Word2vec is designed, as well as the Unified Modelling Language (UML) diagrams to depicts the system design. The Feature extraction tool using Word2vec is developed using Python programming language. The output of this phase is the system design, which is presented through UML diagrams.

3.3 Implementation Phase

In the implementation phase, development of the proposed tool will be carried out using main programming language, Python. The coding will be written using Visual Studio Code and the database will be developed using MySQL Workbench. The software required is Visual Studio Code and MySQL Workbench, whereas the hardware requires an Apple M1 Chip processor, and 8.00GB memory RAM. tool will be tested after development. The output of this phase is the developed tool.

3.4 Maintenance Phase

The maintenance phase will be done using documentation. Test plans are done and completed in this phase. Future works that could be done on the developed tool will be recorded. User manual will be provided for users to refer, which will be written in a readme file. The user manual will guide users to use the tool wisely. The output of this phase is the readme file and the result of testing plans.

4. Analysis and Design

This section explains the actions done in analysis and design phase. In this phase, the UML diagrams, functional and non-functional requirements, and test plans are explained in detail.

4.1 Unified Modelling Language (UML)

The UML diagrams are used to visualise the interaction between users and Feature Extraction Tool, and the system's architecture. The UML diagrams are the use-case diagram, sequential diagram, and activity diagram. Figure 2 visualise the use-case diagram of user and admin. Admin can perform actions such as Login, Manage User, Logout, Update Tool, and Monitor Log. User can perform Register, Login, Upload, Download, and Logout.

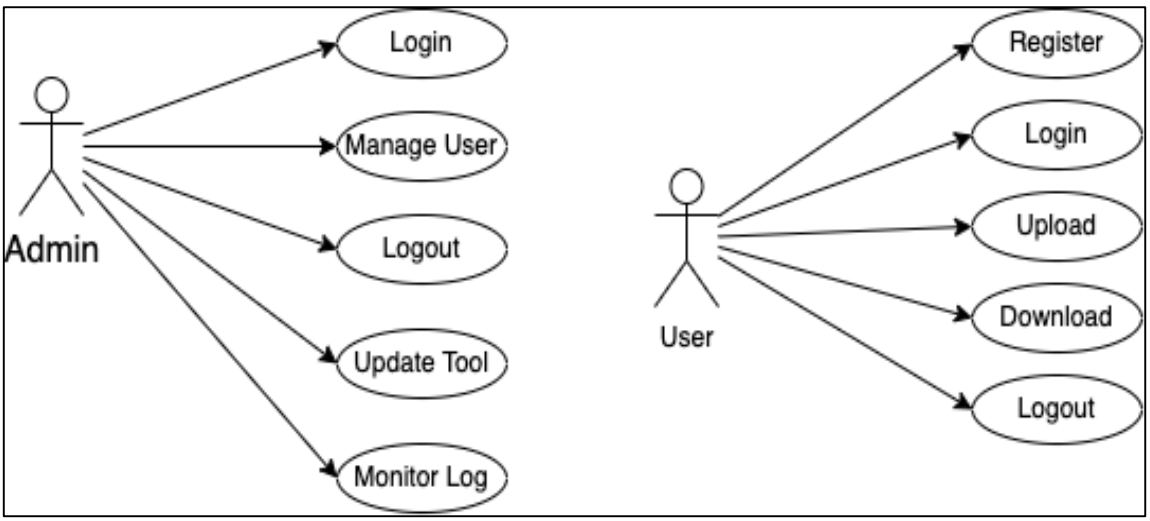

Figure 2: Use-Case Diagram of Feature Extraction Tool for Phishing Dataset

4.2 Sequential Diagram

Sequential diagram shows the order of interaction between users and system. Figure 3 illustrates the sequential diagram of the proposed feature extraction tool. The GUI displays a login page. If a user does not have an account, they can click on the registration button. Once the user submits the registration form, the information will be saved in the database. User will be directed to the home page of the GUI. From there, the user can upload a file for feature extraction, and the file path will be stored for future reference. Once the extraction process is complete, the user will be redirected to the download page. On the other hand, the admin will be redirected to the login page. After logging in, the admin dashboard will be displayed, providing options to update file types, view user lists, or access security logs. The login module is essential to differentiate between normal users and admins. The Register module is exclusively for new normal users, while new admins can only login once an existing admin adds them new admins into the database.

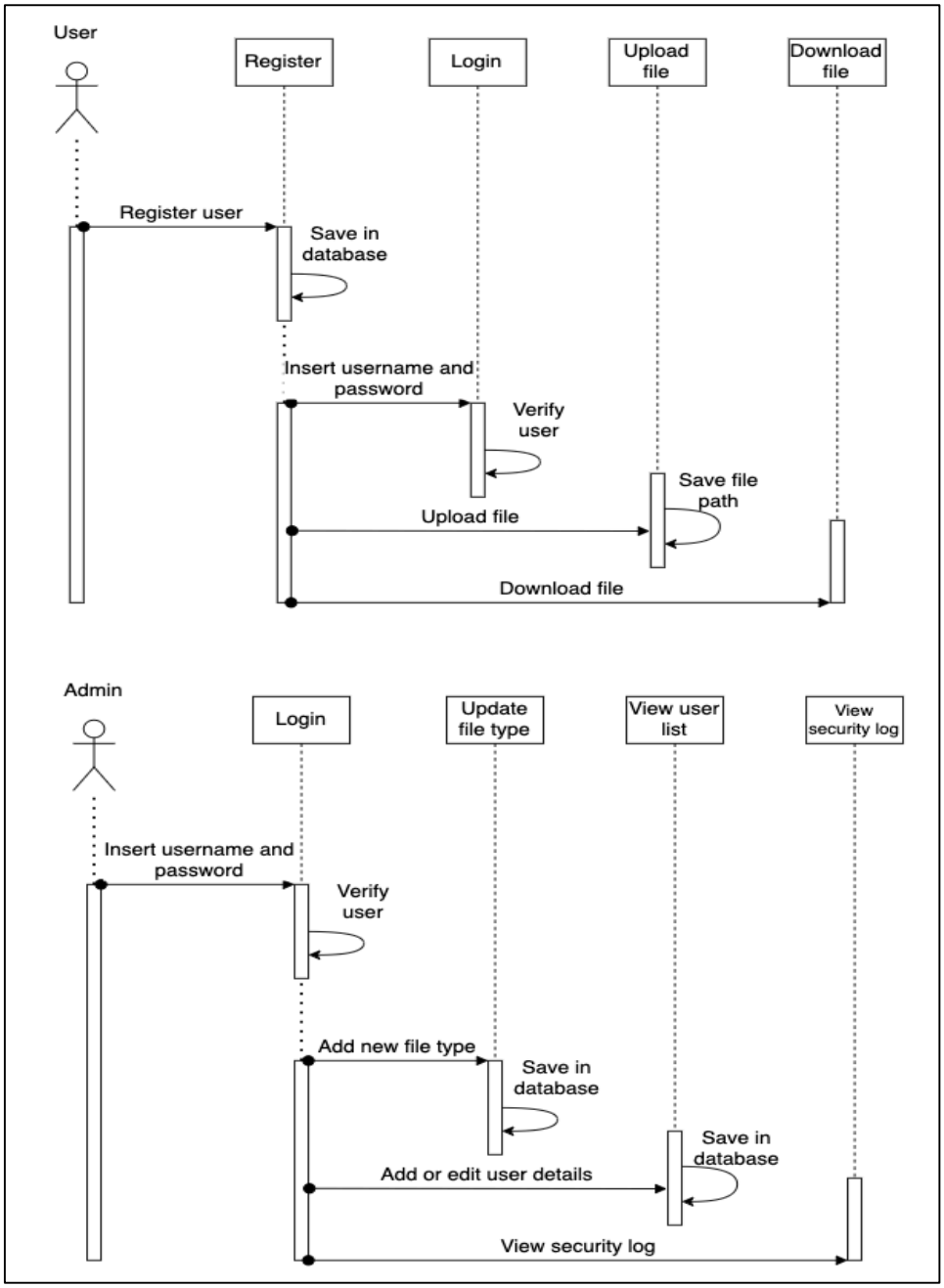

Figure 3: Sequential Diagram of Feature Extraction Tool for Phishing Dataset

4.3 Activity Diagram

Activity diagram is a flowchart which shows the activities that could be done by users in a system. Figure 4 illustrates the activity diagram of user's interaction with the Feature Extraction Tool for Phishing Dataset using Word2vec. If users do not have an existing account, they will be prompted to register a new account. upon successfully login, the home page of the proposed tool will be displayed. The tool utilized the Word2vec model to extract features once the user uploads a raw text file. Subsequently, users are allowed to download the features extracted in a file.

Word2vec is applied in the "Extract Features". A Word2vec model will be built once the user has uploaded the file. The model contains vectors that carry semantic meanings of each word. Each word is converted into vector and is saved into the model for further application in extracting features.

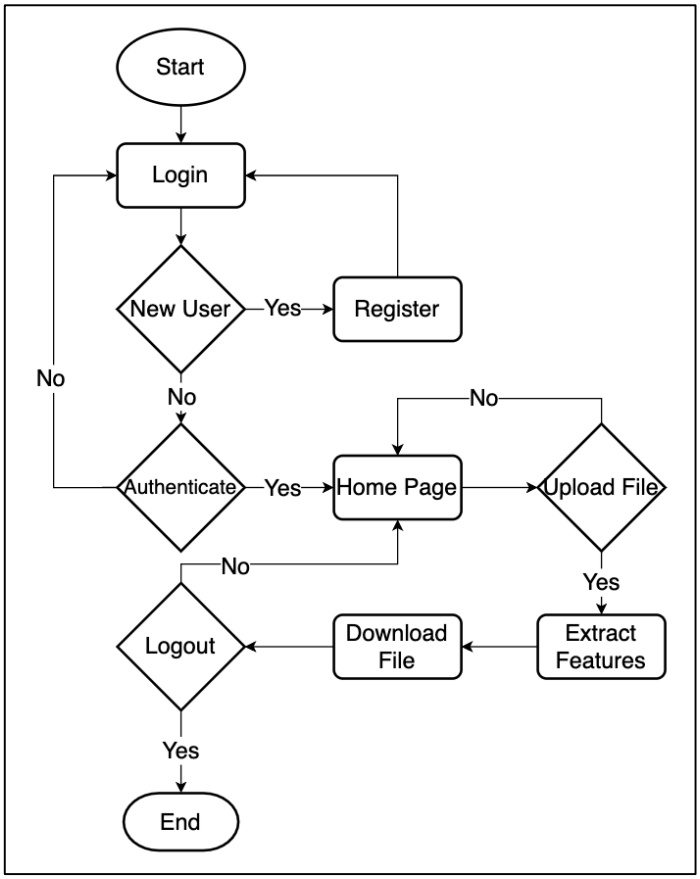

Figure 4: User Activity Diagram of Feature Extraction Tool for Phishing Dataset

Figure 5 illustrates the admin activity diagram using Feature Extraction Tool for Phishing Dataset using Word2vec. Admins possess special functions, including the ability to update the tool by adding new file types for upload, managing user accounts (adding or editing users and new admins), and monitoring the security log.

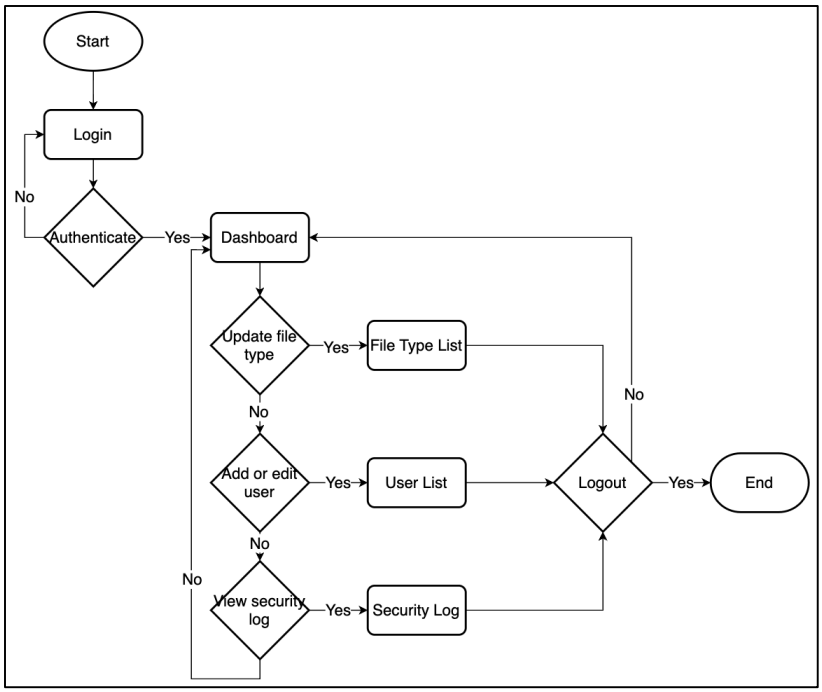

Figure 5: Admin Activity Diagram of Feature Extraction Tool for Phishing Dataset

4.4 Functional Requirements

Table 2 describes the functions of each module. The main modules found in Feature Extraction Tool are upload file module, download file module, feature extraction module, login module, and registration module. The functionalities of each module are explained in the table.

Table 2: Functional Requirements for Feature Extraction Tool for Phishing Dataset

Module	Functionality
Upload File	• User can upload csv file from computers.
Download File	• User can download file after extraction is done.
Feature Extraction	• User can extract features after uploading the file and choosing the features.
User Login	• User can login by inserting the correct username and password if they have created account before. • User are denied to login if incorrect username or password is entered.
User Registration	• New user is required to register a new account using email, username, and password.

Table 3 explains the non-functional requirements for Feature Extraction Tool, which explains the minimum requirements of the tool in the aspect of its performance, operations, usability, and security.

Table 3: Non-Functional Requirements for Feature Extraction Tool for Phishing Dataset

Requirement	Description
Performance	• The tool's function must be able to interact with user.
Operational	• The tool is available without internet connection.
Usability	• The tool is easy to be used by user and has simple user interface design.
Security	• User needs to provide correct username and password to login. • Strong and complex password is required to register a new account.

4.5 Test Plan

Table 4 shows the test plan of Feature Extraction Tool. The test plan will be done in Result and Discussion after development of the tool. This is used to check if the tool has achieved the requirements and provide the required functions.

Table 4: Test Plan for Feature Extraction Tool

Test Category	Description	Expected Result	Actual Result
Admin	Update new file type	Add new file type successfully	Pass / Fail
Admin	View security log	Security log is displayed	Pass / Fail
Admin	Delete user	Selected user is deleted from database	Pass / Fail
User	Register new account: I. Insert new username and password. II. Click on Register button.	Able to login to new account after registration	Pass / Fail
User	Upload file	Able to upload raw text file in csv file	Pass / Fail
User	Extract features	Able to get result after clicking on Extract Features button	Pass / Fail

Table 5 shows the security checklist for Feature Extraction Tool. This checklist is used to ensure that the proposed tool achieves the security requirements stated in planning phase.

Table 5: Security Checklist for Feature Extraction Tool

No	Checklist	Expected Result	Actual Result
1	Make sure that an error message for users shown when the authentication data is wrong. "Incorrect username or password" should be shown instead of "Incorrect username" or "Incorrect password"	"Incorrect username or password" is shown when authentication data is wrong.	Pass / Fail
2	Password validation. Password should have both combination of alphabetic and numeric characters, as well as special characters	User can only register account by providing password that meet the requirements.	Pass / Fail
3	Password should have minimum length of eight characters	User can only register account by providing password that meet the requirement.	Pass / Fail
4	Password should be hidden in textboxes	Password is not visible in textboxes.	Pass / Fail

5. Result and Discussion

This section explains about the implementation phase and maintenance phase of the proposed tool. The tool is developed in the implementation phase, Python is used as the main programming language in Visual Studio Code while completing the development. Test plan and security measurement are checked in maintenance phase after the development of the tool is done.

5.1 System Implementation

The GUI is designed and developed using Python Tkinter library. Figure 6 shows the register and login interface that user and admin will encounter upon opening the tool. Users are required to either log in or register new account for new users to begin using the tool.

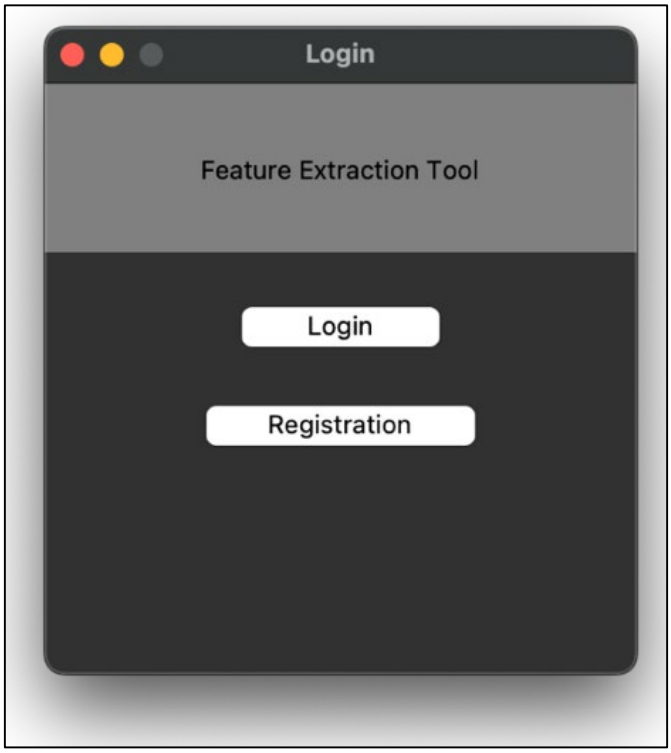

Figure 6: Register and Login Interface Design

Figure 7 presents the password validation code. To create an account, users are required to choose a complex password that meets the following criteria: it must contain a minimum of eight characters, including at least one numeric character, at least one special character, and alphabetic characters. This requirement ensures that users do not set easily guessable passwords, thereby enhancing their account security.

```
def validate_password(password):
    if len(password) < 8:
        messagebox.showerror("Password Error", "Your password should have at least 8 characters")
        return False
    if not any (char.isdigit() for char in password):
        messagebox.showerror("Password Error", "Your password should contain numeric character(s)")
        return False

    special_char = "!@#$%^&*()_-+={}[]|:;<,>.?/~"
    if not any(char in special_char for char in password):
        messagebox.showerror("Password Error", "Your password should contain special character(s)")
        return False

    return True
```

Figure 7: Python Source Code for Password Validation

Figure 8 shows the code of hashing password. All passwords are hashed using SHA-256 encryption with combination of salt. Each user will have different salt, which is their username. Hence, this made the password encryption more complex compared to only hash password using SHA-256 without adding salt [15]. The hashlib module is used to call SHA-256 algorithm. The password and salt will be encoded to bytes and merged to pass into the sha256() function, then it will be converted into hexadecimal format using hexdigest().

```
def hash_password(password, salt):
    hashed_pass = hashlib.sha256(password.encode()+salt.encode()).hexdigest()
    return hashed_pass
```

Figure 8: Python Source Code for Hashing Password

Figure 9 shows the main interface of feature extraction. The user should choose the file, and select features to be extracted, then click the “Preprocess & Extract” button to start.

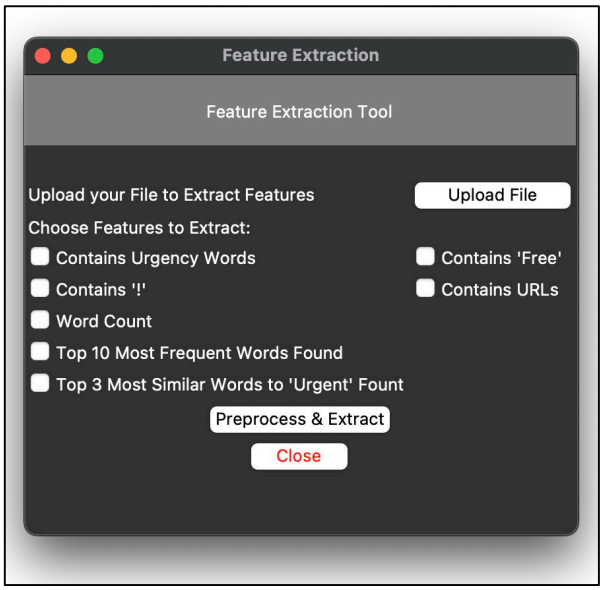

Figure 9: Feature Extraction Interface Design

The proposed tool provides various of features that could be extracted from the raw data. The features and corresponding explanation are shown in Table 6.

Table 6: Features and Explanation

Features	Explanation
Contains_Urgency	Urgency words, which are 'urgent', 'hurry', 'fast', 'verification required', 'important', 'action required', 'now'. This feature will show if the data contains urgency words mentioned above.
Contains_Free	This feature shows if the word 'free' is found in the data.
Contains_Exclamation_Mark	This feature shows if the '!' is found in the data.
Contains_Urls	This feature checks if 'http' found in the data.
Word_Count	This feature counts the total of words in a text.
Contain_Most_Similar_Words_To_Urgent_*Word*	This feature shows top 3 words found most similar to 'urgent' in Word2vec model. Then the occurrence in text is found.
Contain_Most_Frequent_Word_*Word*	This feature shows top 10 most frequent words found in Word2vec model. Then their occurrences in text are found.

Figure 10 shows the function used to initialize a Word2vec model. The genism library is imported to create a Word2vec model. The text_list is the raw data that get from the file chosen. The model is used in extracting features as shown in Figure 11. For example, words that are similar to "urgent" is found using the model, then the words are used to find the in raw text. Same goes to the top 10 most frequent words found in model.

```
global model
model = gensim.models.Word2Vec(sentences=text_list, min_count=1, vector_size=5, sg=1)
```

Figure 10: Python Source Code for Creating Word2vec Model

Figure 11 shows the implementation of the Word2vec model in self-defined functions. Word2vec model carries the semantic meaning of words and relationship between words in the word corpus from the dataset uploaded by users. Hence, the most similar words to a given word could be found by searching the word using model.wv.most_similar() function.

```
def top_3_most_similar(text):
    similar_words = model.wv.most_similar('urgent', topn=3)
    similar_words_list = []
    for word, vector in similar_words:
        similar_words_list.append(word)

    for i in range(3):
        result = []
        for row in text:
            if similar_words_list[i-1] in row:
                result.append('1')
            else:
                result.append('0')
        df['Contain_Most_Similar_Words_To_Urgent_' + similar_words_list[i-1]] = result

def top_10_most_freq(text):
    most_frequent_list = []
    for index, word in enumerate(model.wv.index_to_key):
        if index == 10:
            break
        most_frequent_list.append(word)

    for i in range(10):
        result_most_freq = []
        for row in text:
            if most_frequent_list[i-1] in row:
                result_most_freq.append('1')
            else:
                result_most_freq.append('0')
        df['Contain_Most_Frequent_Word_' + most_frequent_list[i-1]] = result_most_freq
```

Figure 11: Python Source Code for Extracting Features Using Word2vec Model

In the proposed tool, top 3 most similar words to the word “urgent” are selected. This is because the similar words may carry similar emotions and meanings that are important to detect phishing. Besides that, the Word2vec model generates the most frequent words used in the word corpus too. Hence, users could find out what are the most frequent words used in the dataset. If the dataset contains more phishing text, user may find out that some of the most frequent words used in phishing text, and the model could show how many words from the list could be found in each content.

The extracted features will be saved as a new csv file automatically to the same file path as the uploaded file after extraction is done since the file is created before appending the result into the file. Figure 12 shows the code used to create a csv file.

```
feature_file = outputname +'_features.csv'
text = file.iloc[:, 0]

if(os.path.exists(feature_file) and os.path.isfile(feature_file)):
    os.remove(feature_file)

dataframe = pandas.DataFrame()
dataframe.to_csv(feature_file)
```

Figure 12: Python Source Code for Creating CSV File

Hence, a table of the result will be displayed after extraction is done as a proof of the extraction is completed. Figure 13 shows the code for displaying the result, and Figures 14 and 15 show the GUI of displaying both the pre-processed file and the extracted features file.

```
def display(dataframe):
    global root2
    root2 = Toplevel(root)
    root2.title('Extracted Features')
    root2.geometry("700x500")
    root2 = ps.Table(root2, dataframe=dataframe, showtoolbar=False, showstatusbar=False)
    root2.show()
```

Figure 13: Python Source Code for Displaying Result

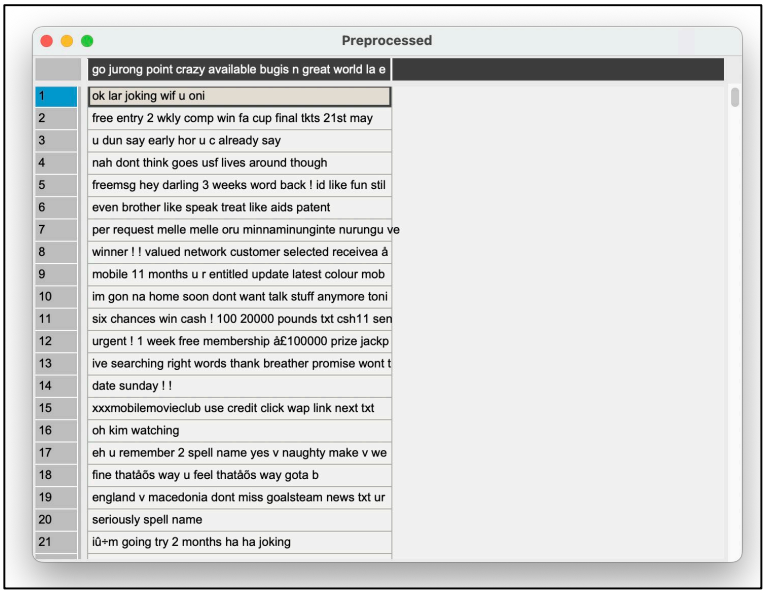

Figure 14: GUI for Displaying the Pre-processed Dataset in CSV File

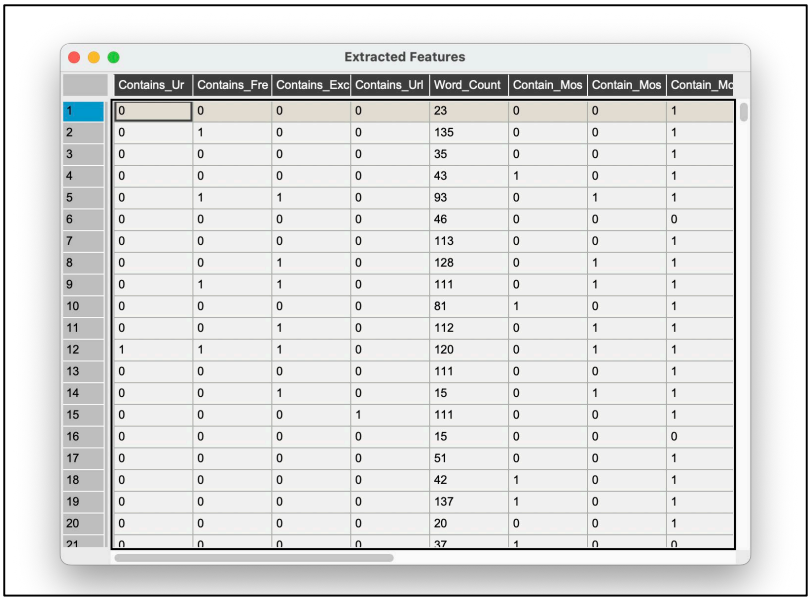

Figure 15: GUI for Displaying the Extracted Features in CSV File

The extracted features will be saved to the file path of the raw dataset once the extraction is completed. Figure 16 shows that the data is saved as csv file to the file name defined as feature_file in Figure 12. The file name consists of file path from the uploaded file and the file name.

```
df.to_csv(feature_file, index=False)
display(df)
```

Figure 16: GUI for Displaying the Extracted Features

5.2 System Testing and Verification

This section explains and shows how the Feature Extraction Tool is tested in its functions, security measures, and user acceptance. This is to ensure that the tool is developed according to the requirements and objectives mentioned in the planning phase.

5.2.1 System Functionality and Security Test

Table 7 and Table 8 shows the test plan that are carried out after the development is done. The test plan includes both the functionality test plan and the security test plan. Table 7 is the test plan conducted on user and admin. It tested the functionality of the Feature Extraction Tool, and the result is pass if the expected result is shown. Feature Extraction Tool has passed all the functionality test.

Table 7: Functionality Test Plan

Test Category	Description	Expected Result	Actual Result
Admin	Update new file type	Add new file type successfully	Pass
Admin	View security log	Security log is displayed	Pass
Admin	Delete user	Selected user is deleted from database	Pass
User	Register and login new account	Able to login to new account after registration	Pass
User	Upload file	Chosen csv file is selected and file name is displayed	Pass
User	Extract features	Extracted features is shown after extraction	Pass

Table 8 shows the security test plan of the Feature Extraction Tool. This test plan is conducted to check if the tool has reached the expected result of the security measurements listed in the test plan. In this case, Features Extraction Tool has given the expected results, hence the results are passed.

Table 8: Security Test Plan

No	Checklist	Expected Result	Actual Result
1	Make sure that an error message for users shown when the authentication data is wrong. "Incorrect username or password" should be shown instead of "Incorrect username" or "Incorrect password"	"Incorrect username or password" is shown when authentication data is wrong.	Pass
2	Password validation. Password should have both combination of alphabetic and numeric characters, as well as special characters	User can only register account by providing password that meet the requirements.	Pass
3	Password should have minimum length of eight characters	User can only register account by providing password that meet the requirement.	Pass
4	Password should be hidden in textboxes	Password is not visible in textboxes.	Pass

5.2.2 User Acceptance Test

The proposed tool is tested in terms of user acceptance and functionality. For test plan result, both functionality and security are passes. Table 9 shows the user acceptance form given to users, and the Figure 17 shows the result of the form. A total of 10 respondents from Universiti Tun Hussein Onn Malaysia response the form.

Table 9: User Acceptance Form

Question	Acceptance Requirement	Result	
		Yes	No
1	The tool must execute from start to end.		
2	User able to register and login to the account.		
3	User able to upload file.		
4	User able to pre-process the uploaded file.		
5	User able to extract selected features from the uploaded file.		
6	The tool is user-friendly.		
7	Admin able to register and login to the account.		
8	Admin able to view and delete user account.		
9	Admin able to view security log		
10	Admin able to view and add new file type.		

All respondents give positive result for Question 1 to Question 10 based on Figure 17. This shows that users can successfully register or login to start using the Feature Extraction tools using Word2vec. They can choose and upload the file and preprocess the selected file. Next, users can choose desired features as listed in the tools to be extracted from the raw data. For admins, they are able to manage user list, and to view security log. Besides that, admins could add file type to allow users to choose more types of files as a future improvement work.

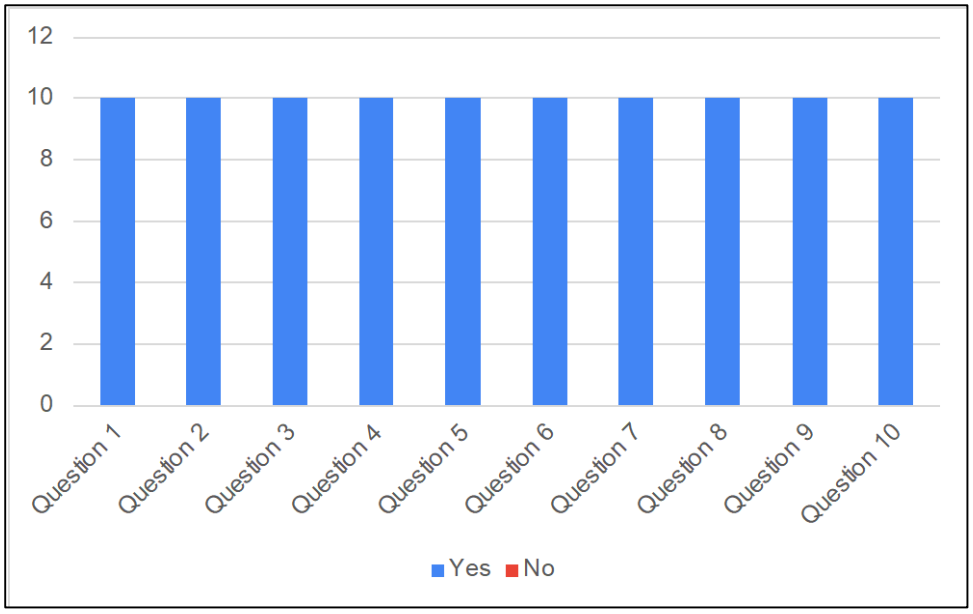

Figure 17: User Acceptance Result in Google Form

6. Conclusion

The proposed tool has accomplished all objectives listed, which are to design a feature extraction tool to extract phishing dataset, to develop a feature extraction tool by applying Word2vec model, and to test the feature extraction tool with phishing dataset in terms of user acceptance and system functionality.

There are some advantages and disadvantages found in the tool. The advantages of the tool are the tool is user-friendly which has simple button and instructions to guide users to use the tool from start to end, it allows users to choose features that they want to extract only, and also preview the output after preprocess and extraction are done. Besides that, the tool protects the users' password using SHA-256 encryption with salt. The features provided are Contains_Urgency, Contains_Free, Contains_Exclamation_Marks, Contains_Urls, Word_Count, Contain_Most_Similar_Words_To_Urgent_Word, and Contain_Most_Frequent_Word_Word. The disadvantages are the tool only allows users to upload csv file at the moment, and it has limited features that could be selected.

Future works are needed to improve the tool and make it more useful in all manner. The tool should allow users to upload files regarding any file type. Besides that, more features should be added in future so that it could extract more features that are helpful in machine learning. This tool is expected to help in future research works.

Acknowledgment

The authors would like to thank the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia for its support.

References

[1] Anti-Phishing Working Group (APWG), "Phishing Activity Trends Report, 3rd Quarter 2022," Dec. 2022. Available: https://docs.apwg.org/reports/apwg_trends_report_q3_2022.pdf?_gl=1*ctw1ti*_ga*MTY4MzgwOTIzOC4xNjcxNjgzOTMw*_ga_55RF0RHXSR*MTY3MjY0NDExMy4zLjAuMTY3MjY0NDENDE4wLjAuMA..&_ga=2.30539259.513969933.1672644114-1683809238.1671683930. [Accessed Oct. 27, 2022].

[2] F. D. Michael Dass and C. F. Mohd Foozy, "A Comparative Study of SQL Injection Detection Using Machine Learning Approach," *Applied Information Technology And Computer Science*, vol. 3, no. 2, pp. 19–31, Nov. 2022, [Online]. Available: https://publisher.uthm.edu.my/periodicals/index.php/aitcs/article/view/7370. [Accessed Oct. 27, 2022]

[3] M. Zareapoor and S. K. R, "Feature Extraction or Feature Selection for Text Classification: A Case Study on Phishing Email Detection," International Journal of Information Engineering and Electronic Business, vol. 7, no. 2, pp. 60–65, Mar. 2015, doi: 10.5815/ijieeb.2015.02.08.

[4] A. Przepiórkowski, M. Piasecki, K. Jassem, and P. Fuglewicz, *Computational Linguistics: Applications*, vol. 458. Springer, 2012. [E-book] Available: Springer.

[5] M. Sammons, C. Christodoulopoulos, P. Kordjamshidi, D. Khashabi, V. Srikumar, and D. Roth, "EDISON: Feature Extraction for NLP, Simplified," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, Portorož, Slovenia: European Language Resources Association (ELRA), May 2016, pp. 4085–4092. [Online]. Available: https://aclanthology.org/L16-1645

[6] (2019) "YAKE! Keyword Extractor," *Ricardo Campos*. Retrieved from Google Play Store. Available: https://play.google.com/store/apps/details?id=com.yake.yake

[7] T. Mikolov, W. Yih, and G. Zweig, "Linguistic Regularities in Continuous Space Word Representations," in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Atlanta, Georgia: Association for Computational Linguistics, Jun. 2013, pp. 746–751. [Online]. Available: https://aclanthology.org/N13-1090

[8] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," in *NIPS'13: Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, Oct. 2013, pp. 3111–3119.

[9] K. R. Chowdhary, "Natural Language Processing," in *Fundamentals of Artificial Intelligence*, New Delhi: Springer India, 2020, pp. 603–649. doi: 10.1007/978-81-322-3972-7_19.

[10] M. Zubair Asghar, A. Khan, S. Ahmad, and F. Masud Kundi, "A Review of Feature Extraction in Sentiment Analysis," *Article in Journal of Basic and Applied Research International*, vol. 4, no. 3, pp. 181–186, 2014, [Online]. Available: www.textroad.com. [Accessed Oct. 27, 2022].

[11] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," Jan. 2013, doi: 10.48550/arXiv.1301.3781.

[12] RED-Alert, "Linguistic feature extraction tool," 2018. Accessed: Nov. 20, 2022. [Online]. Available: http://redalertproject.eu/linguistic-feature-extraction-tool/

[13] Naheem Noah, Abebe Tayachew, Stuart Ryan, and Sanchari Das, "PhisherCop: Developing an NLP-Based Automated Tool for Phishing Detection," University of Denver, 2022.

[14] F. Natasha Baharudin and K. Malik Mohamad, "NPC-WIPER: File Wiper Tool using Non-Printable Characters," *Applied Information Technology And Computer Science*, vol. 3, no. 2, pp. 197–208, 2022, doi: 10.30880/aitcs.2022.03.02.013.

[15] W. Buchanan, *Cryptography*. River Publishers, 2022. [E-book] Available: Google Books.