# CONTEXT-SENSITIVE KEYWORD DENSITY BASED SUPERVISED LEARNING TECHNIQUES FOR DETECTION OF MALICIOUS WEB PAGES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

BETÜL ALTAY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

AUGUST 2016

Approval of the thesis:

**CONTEXT-SENSITIVE KEYWORD DENSITY BASED SUPERVISED LEARNING TECHNIQUES FOR DETECTION OF MALICIOUS WEB PAGES**

submitted by **BETÜL ALTAY** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Gülbin Dural Ünver _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Adnan Yazıcı _____
Head of Department, **Computer Engineering**

Prof. Dr. Ahmet Coşar _____
Supervisor, **Computer Engineering, METU**

Asst. Prof. Tansel Dökeroğlu _____
Co-supervisor, **Computer Engineering, UTAA**

**Examining Committee Members:**

Prof. Dr. Halit Oğuztüzün
Computer Engineering Department, METU _____

Prof. Dr. Ahmet Coşar
Computer Engineering Department, METU _____

Asst. Prof. İsmail Sengör ALTINGÖVDE
Computer Engineering Department, METU _____

Assoc. Prof. Murat Manguoğlu
Computer Engineering Department, METU _____

Assoc. Prof. Murat Koyuncu
Computer Engineering Department, Atılım University _____

**Date:** _____

**I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.**

Name, Last Name: BETÜL ALTAY

Signature :

# ABSTRACT

CONTEXT-SENSITIVE KEYWORD DENSITY BASED SUPERVISED LEARNING TECHNIQUES FOR DETECTION OF MALICIOUS WEB PAGES

Altay, Betül

M.S., Department of Computer Engineering

Supervisor : Prof. Dr. Ahmet Coşar

Co-Supervisor : Asst. Prof. Tansel Dökeroğlu

August 2016, 64 pages

Conventional methods use a black list in order to decide whether a web page is malicious or not. These black lists are generally produced by technicians or operators and used for the security purposes of the organizations, protection of software from web based virus attacks, web browsers, etc. However, the black-list approach is not a scalable solution for the frequently changing and rapidly growing number of web pages on the internet and their dynamic contents. In this thesis, we propose and analyze a method for the classification of the web pages by using Support Vector Machine, Maximum Entropy, and Extreme Learning Machine techniques. The performance of the proposed machine learning models are evaluated with 100K web pages. Features of web pages are generated by processing HTML contents and information is obtained using conventional feature extraction methodologies, such as existence of words, keyword frequencies, and a novel method based on keyword densities. The performances of machine

learning methods employing various extracted features are analyzed and experimental results show that the proposed method can identify malicious web pages with a very high accuracy of up to 98.24% while also achieving practical web page processing times.

Keywords: Machine Learning, Malicious Web Pages, Binary Classification, Keyword Density

# ÖZ

İÇERİK-DUYARLI ANAHTAR KELİMELERE DAYALI GÖZETİMLİ ÖĞRENME TEKNİKLERİYLE ZARARLI WEB SİTESİ TESPİTİ

Altay, Betül
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. Ahmet Coşar
Ortak Tez Yöneticisi : Yrd. Doç. Tansel Dökeroğlu

Ağustos 2016 , 64 sayfa

Web sayfalarının zararlı olup olmadıklarına karar verilmek için genellikle kara listeler kullanılmaktadır. Bu listeler teknisyen veya operatörlerin her bir web sitesinin zararlı olup olmadığına karar verip, zararlı görünüyorsa bu listelere eklemesi ile oluşturulurlar. Ardından, bu listeler virüs koruma programları, web tarayıcılar ve çeşitli özelleşmiş ürünlerle bireylerin ve kurumların güvenlik sorunlarına çözüm getirmek için kullanılırlar. Ancak, hızla değişen ve büyüyen web sitesi sayısı ve içerikleri düşünüldüğünde bu yaklaşım ölçeklenebilir bir çözüm getirememektedir. Bu tez çalışmasında, Support Vector Machine, Maximum Entropy ve Extreme Learning Machine teknikleri kullanılarak web sayfalarının sınıflandırılması üzerine bir yöntem tasarlayıp analiz etmekteyiz. Bu makine öğrenimi modellerinin performansları yüz bin web sitesi örneğiyle bulunup karşılaştırılmaktadır. Web sayfalarının özellikleri HTML içerikleri kullanılarak hazırlandılar. Bu özellikler geleeksek özellik çıkarma yöntemleri olan kelimelerin

içerikte bulunmasına dayanan ikili gösterim, anahtar kelime sayısı ve yeni bir yöntem olan anahtar kelime yoğunluğu ile ifade edildiler. Önerilen makine öğrenimi yöntemlerinin performansları analiz edildi. Deneysel sonuçlar, önerilen yöntemlerin web sayfalarının zararlı olup olmadıklarını uygun sürelerde çalışmaları sonucunda %98.24 oranına varan doğruluk oranı ile belirleyebildiklerini göstermiştir.

Anahtar Kelimeler: Makine Öğrenimi, Zararlı Web Sayfaları, İkili Sınıflandırma, Anahtar Kelime Yoğunluğu

To my family

# ACKNOWLEDGMENTS

I would like to thank my supervisor Professor Ahmet Coşar and co-supervisor Assistant Professor Tansel Dökeroğlu for their support and guidance.

Also, sincerest thanks to each of my family members for supporting, motivating and believing in me all the way through my life.

# TABLE OF CONTENTS

# LIST OF TABLES

TABLES

# LIST OF FIGURES

FIGURES

# LIST OF ABBREVIATIONS

| | |
|---|---|
| ANN | Artificial Neural Network |
| DB | Database |
| DT | Decision Tree |
| ELM | Extreme Learning Machines |
| KD | Keyword Density |
| KNN | K-Nearest Neighbor |
| LBFGS | Limited-Memory Variable Metric |
| MaxEnt | Maximum Entropy |
| ME | Maximum Entropy |
| ML | Machine Learning |
| OWLQN | Orthant Wise Limited-memory Quasi Newton |
| RBF | Radial Basis Function |
| SGD | Stochastic Gradient Descent |
| SLFN | Single-hidden Layer Feed-forward Networks |
| SVM | Support Vector Machine |
| TF-IDF | Term Frequency Inverse Document Frequency |

# CHAPTER 1

# INTRODUCTION

## 1.1 Malicious web pages

In accordance with the statistics of International Telecommunication Union (ITU) [10], the number of individuals using the internet is 3.17 billions over the world in 2015. Moreover, the internet usage has become essential for our common daily activities such as shopping, education, entertainment, keeping and managing private information, banking and social networking. Unfortunately, the huge usage of the internet and its facilities cause great danger in security because cyber criminal activities have become easier. Web pages including threats for users are called Malicious Web Pages. On the other hand, innovative technologies in web design has also been improved. In the past, web pages were including only static HTML contents but nowadays they include technologies giving opportunities user interaction. This situation also causes significant gaps in online security. The most important security threats included in web pages are called Phishing and Cross Site Scripting.

*Phishing*, also called as *Fishing*, is an attempt to obtain personal information of internet users by using social engineering 1.1. Stolen information may include user names, passwords, e-mail addresses, phone numbers, photos, social security numbers and even credit card details of victims [11]. These information about users are used for advertising or crimes such as stealing money. The operation mechanism of phishing is formed with faking. For example, web pages may include fake links. These links can download a harmful executable to users'

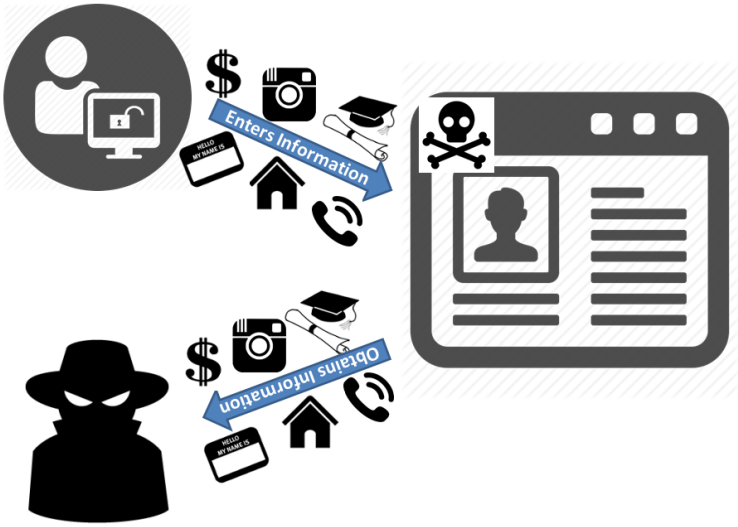


Figure 1.1: Phishing. Adapted from [1] [2] [3] [4] [5] [6] [7]

computers. Also, the link can open another malicious or unwanted pages such as gambling or sexual malicious documents. For the second example, web pages may act like known bank or government agency web sites. After that, this page requests from user entering personal information and user could be deceived. Lastly, users are exposed to fake advertising and counterfeit products selling because of the fake web pages. After the user buy a product, it may be imitation, illegal or even an empty box.

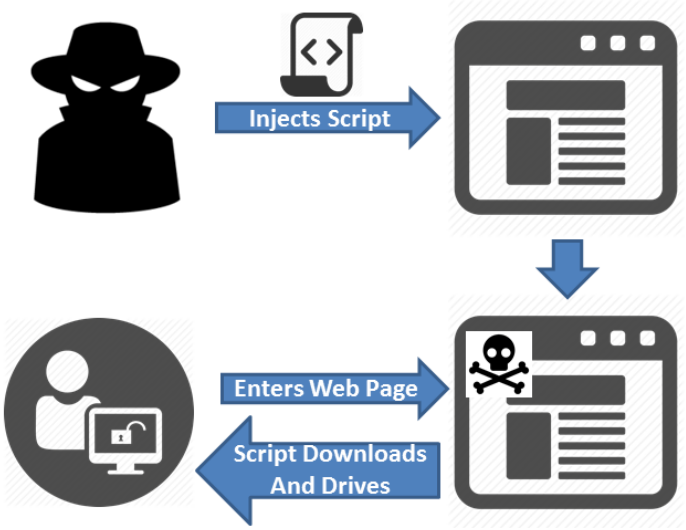


Figure 1.2: Cross Site Scripting. Adapted from [8] [2] [3] [9] [5]

*Cross Site Scripting*, also called as *XSS*, is a type of computer security vulnerability found in web. XSS gives opportunity to attackers injecting malicious code into web pages 1.2. After injection, a victim's browser and machine becomes vulnerable while visiting the web page [12]. It had become a major issue especially after improving web page design and creation of scripting technologies in web pages. At the beginning of web page design, they were created only with static HTML. Nowadays, web developers use dynamic technologies such as Javascript, ActiveX, Silverlight and Java Applets. Those drive-by-download techniques make the service easier, powerful and flexible. For example, users read PDF files on browsers through ActiveX. However, power and flexibility of recent web pages bring a new tool for attackers by increasing misuse opportunities. Recent development tools such as Adobe Flash, Java Script, Visual Basic and PHP have ability of downloading and executing any code through the internet [13]. For an example scenario of XSS, a user visits an e-commerce web site and buys a product with credit card. After entering information, a criminal person may retrieve privacy information about user by using cookies. Although some scripts such as loading image and iFrames are represented to be safe, they may run additional malicious codes during load an image or another web page. Some studies show that there is a large number of malicious web pages in search results [14]. Because of the threats' importance, variety and common, filtering malicious web pages becomes absolutely essential.

## 1.2 Machine learning and web page classification

In order to handle the explained problem above, various solutions are proposed and used. Firstly, browsers and security tools have blacklists including malicious web domains and URLs. For example, if requested URL is found on the blacklist of Google Safe Browsing, Google browser does not crawl the page. However, blacklisting approach has some deficiencies; (1) the lists only include crawled web pages earlier, (2) their crawlers could not reach intranets, (3) crawled pages may be hacked later, (4) they also need a malicious page detection mechanism or human resources on composing of the lists process [15]. Second method is creat-

ing honeypots with Virtual Machines(VM). By using VM environments, visiting a web page is simulated so its effects may be observed. It is a successful method but not efficient with its processing time. Therefore, this method may help creation of blacklists but it is not usable with real time classification processing [16]. Third method is signature check. However, signature check is implemented only executable instead of phishing or scripting. Also, its performance is not good [17] [18]. Lastly, some studies have been done on automated solutions with machine learning. These methods generally use counts of static features of web pages in order to reduce feature count. In this thesis, we provide an automated method for detecting malicious web pages based on supervised machine learning (ML) algorithms with keyword densities of HTML content because the content of a web page is easily reachable and also it has potential information to detect its safety. Hereby, results of this study shows better accuracies than the others.

### 1.3 Structure of the thesis

The outline of the thesis is as follows. In chapter 2, we explain *related works* about malicious web page detection studies and supervised machine learning techniques. In chapter 3, we introduce *machine learning models* used in this study. These supervised machine learning methods are Support Vector Machine (SVM), Maximum Entropy (ME) and Extreme Learning Machine (ELM) because they are recent and successful methods in similar studies. In chapter 4, we provide information about *experimental setup* of this study. This chapter describes preparing data steps such as crawling data sets, keyword extraction, database usage, feature sets and feature vectors. In chapter 5, *evaluation* of the proposed ML techniques are presented. In order to obtain detailed information about outputs, different parameters and conditions are tested separately. Finally, we give our concluding remarks and future work in chapter 6.

# CHAPTER 2

# RELATED WORK

In this chapter, we give information about the previous related studies and existing approaches to detect malicious web pages by using machine learning(ML) techniques on web content.

This subject combines utterly diverse concepts which have different evolution processes, usage areas, histories and types of effects on the topic of this study. We tried to explain related works of each sub topic clearly and separately in four parts;

- malicious web page identification and detection without ML techniques,
- document and web page classification (web filtering) using ML with content features,
- malicious web page detection using ML with non-content features,
- malicious web page detection using ML with content features that is the most related one with this thesis study.

## 2.1 Malicious Web Page Identification and Detection without Using Machine Learning Techniques

Chen and Guo showed phishing (fishing) had emerged in 1990s as a recent type of network attack of web page which cheats users in order to reach users' personal information [19]. They suggested an end-host based anti-phishing algorithm

to detect and prevent phishing attacks by using generic characteristics of the hyperlinks. They successfully detected 195 out of 203 phishing e-mail attacks with their method. Next year, Moshchuk, Bragin, Deville, Gribble and Levy designed a proxy-based anti-malware tool which uses Virtual Machines (VM) [20]. User reaches to web page after the tool renders it in a VM. Execution-based detection was a new approach instead of signature control of other anti-malware tools. Invernizzi, Comparetti, Benvenuti, Kruegel, Cova and Vigna studied in a different perspective of finding malicious web pages [21]. Their aim was not filter web pages in client side on run time as usual. They focused on crawling malicious web pages. Therefore, they searched the web by starting from a known malicious web page and crawled only malicious ones by comparing with initial seed. These studies and approaches are not directly related with our study on algorithms and evolution step. However, they put forward the importance of malicious web page problem and gives some different perspectives to solve the same problem.

## 2.2 Document and Web Page Classification (Web Filtering) Using Machine Learning with Content Features

Malicious web page detection may be thought as the sub class of document classification and web page filtering. These subjects include much more search and implementation areas because they were seen in earlier years. There are too many papers related with these topics but we chose to focus three of them because of high correlation with our study. Nigam, Lafferty and McCallum proposed maximum entropy usage in text classification because of its usage in similar works which are language modeling, part-of-speech tagging and text segmentation [22]. Also, they showed that maximum entropy is a valid technique for text classification by obtaining better results in some conditions on comparison with Naive Bayes which is a successfully used ML technique. Pang, Lee and Vaithyanathan published a research about binary classification of documents and it was attracted attention in academia [23]. This paper compares three successful ML techniques for sentiment classification of HTML documents.

Performances of the ML techniques, which are SVM, MaxEnt and Naive Bayes, was similar. Chau and Chen focused on the search of related web pages and implemented neural network and SVM techniques with content and link features in order to be used for topic specific search engines [24]. They also used HTML tags in order to decide importance of words in a web page document because location of the words in an HTML document gives considerable information. The studies in this sub topic are not directly related with our study by considering the problem definition. However, ML techniques and features of these studies guide us on design of similar classification solution. We extracted the valid ML techniques and importance of HTML tags for calculation of keyword density from studies explained in this paragraph.

## 2.3 Malicious Web Page Detection Using Machine Learning with Non-Content Features

Malicious web sites include another common features as well as contents. There are some related studies which use the count or length of some static features because feature count is small and times of classification are low when counts of something are selected as features. Six DNS and Server relation features are used by Seifert et. al. [25]. These features are numbers of unique HTTP servers, redirects, redirects to different countries, redirects to same country, domain name extensions and unique DNS servers. Decision Tree ML technique is used with almost 18.500 samples and resulted with 74.5% true positive rate and 97.4% true negative rate. Seifert et. al. also used different eight static features [26]. These features are numbers of applet and object tags, script tags, XML processing instructions, frames and iFrames, indications of redirects, source script tags, functions that indicate script obfuscation, visibility of iFrames. Decision Tree ML technique is used with almost 21.500 samples and resulted with 94.1% true positive rate and 53.8% true negative rate. Then, 171 features are used [27]. 154 of them are counts of native JavaScript functions such as abs(), acos(), apply(), etc. 9 of them are static features of HTML document such as word count, line count, symmetry of tags, etc. 8 of them counts of the use of ActiveX objects

such as Scripting.FileSystemObject, WScript.Shell, Adodb.Stream, etc. Naive Bayes, Decision Tree, SVM and Boosted Decision Tree algorithms were trained and tested with almost 1100 samples and Boosted Decision Tree showed best results with 92.6% true positive and 92.4% true negative rates. Also, 77 static features are used by Canali et. al. [28]. 19 of them are HTML related features such as iFrame tag count, hidden element count, the percentage of white spaces, known malicious pattern count. 25 of them are JavaScript related features such as the number of occurrences of eval(), setTimeOut(), setInterval() functions, number of built-in functions commonly used for obfuscation routine, number of long variable names, number of string modification functions, etc. 33 of them are URL link and host features such as suspicious URL pattern count, presence of IP address, presence of sub-domain, value of TTL, registration date, etc. SVM, Random Tree, Random Forest, Naive Bayes, Logistic, J48 and Bayes Net are used with almost 200.000 samples. At a result, 94.5% true positive and 95.8% true negative rates were obtained. Lastly, 39 static features are used [13]. Ten of them are URL features such as number of words, length of host, etc. 29 of them are page content related features such as applet count, embedded script count, abnormal visibility, style, etc. SVM, Decision Tree, Naive Bayes, KNN and ANN are used with almost 30.000 benign samples. At a result, 96.01% accuracy was obtained with ANN. These studies are related with our study considering in problem and data sets. However, feature set size and characteristics, algorithms and experiment processes are totally different. Therefore, we did not focus these papers or use them directly but they should be analyzed and a hybrid system should be designed if a comprehensive product will be developed for malicious web page detection.

## 2.4 Malicious Web Page Detection Using Machine Learning with Content Features

The most recent and related studies are listed in this section. Bannur, Saul, and Savage used conventional URL features, number of page links, semantic and visual features of web contents [16]. They chose to implement SVM and Logis-

tic Regression methods. At a result, they decreased the error rate to 1.9% with SVM. Abbasi, Zahedi, Kaza and others put forward a similar research with medical web pages [29]. Their study was rich about classification method varieties because they implemented 21 classification methods. Graph-based methods are listed from most successful to least successful as RTL-GC, QoC, Mass Estimation, QoL, TrustDistrust, TrustRank, AntiTrustRank, BadRank, Cautious Surf, PageRank and ParentPenalty. On the other side, content-based methods are RTL-CC, AZProtect, SVM-Linear, Logistic Regression, SVM-RBF, SVM-Poly, Bayes Network, Neural Network, C4.5 and Naive Bayes sequentially. Last related study is done by Kazemian and Ahmed [18]. They used URL, page links, semantic and visual features together as previous studies. They employed three supervised ML techniques such as KNN, SVM and Naive Bayes, and two unsupervised ML techniques such as K-Means and Affinity Propagation. The study proposed that RBF-SVM technique is the best with whole feature value types. The true positive rate was 97.8% and true negative ratio was 55.1% in this study.

## 2.5 Comparison between Related Works and This Study

Non-content features have been preferred generally on malicious web page detection with ML techniques. There are two main reasons; easy data preparation and fast training in ML techniques with small size of features. However, each word of an HTML content gives clues about web site's meaning and behaviors. Therefore, in this study, we focused on content of the web pages. Also, we do not prefer adding other features like URLs, screen-shots, DNS server relationships etc. because these features have totally different characteristics so they may distort the results. On the other hand, by considering the data is not only a text but also a web page, we also used a new methodology for deciding weights of features; keyword density while the other studies use conventional methods; existence or frequency of keywords. In addition to feature modifications, we also chose to work with different ML techniques in this study; Maximum Entropy and Extreme Learning Machine. We chose Maximum Entropy because of the success on document classification in previous studies [22] [23] but it has not

been implemented for any malicious web page detection study. Secondly, Extreme Learning Machine provides faster learning speed and less human interfere than SVM [30]. We chose to work with ELM because learning speed is important in this study because we have approximately 800.000 keyword features and 100.000 web pages. Third ML technique of our study is SVM which has been used in lots of related works and proved its success. Our aim is using it as a base classification technique in order to obtain meaningful comparison with the unpracticed ML techniques; ME and ELM. Lastly, we aim to better accuracy of the results. By increasing effort especially on data processing step, we propose to increase accuracy level because even most recent and similar study [18] provides 97.8% true positive rate with 44.9% false positive.

# CHAPTER 3

# APPLIED MACHINE LEARNING MODELS

In the following sections, there is an overview of the supervised ML techniques that have been applied in this study. The classification algorithms we tried with our data sets are Support Vector Machine (SVM), Maximum Entropy (ME) and Extreme Learning Machine (ELM).

To clarify why we chose these methods, we should explain recent and similar studies at first. These studies, which are stated in section 2.4, shows that SVM performs best results compared to lots of popular classification models such as Logistic Regression, Bayes Network, Neural Network, Naive Bayes, K-Nearest Neighbors, K-Means, Affinity Propagation etc. Therefore, SVM has proven itself on binary classification by producing best results in similar malicious web page detection studies. [16] [18]. Besides, we wanted to show the results of untried methods; ELM and MaxEnt. Although philosophies behind these three supervised ML techniques are quite different, the reasons for using them can be summarized with their efficiencies and similar uses in the previous studies. Maximum Entropy has been shown to be an effective method on text categorization and document classification but it has not been tried for web page classification yet. [23] ELM is a popular classification method and convenient for text classification because of its learning speed but it has not been tried on web content classification as well.

## 3.1 Support Vector Machines

Support Vector Machines (SVM) Model is widely used data classification technique for binary classification of high dimensional data. The concept of support vectors was introduced by Boser et al. [31]. It is a binary classification technique that finds optimal margin between the training patterns and the decision boundary on seperable data. Then, Costes and Vapnik (1995) designed the present and more convenient form of SVM for non-separable data [32]. The main target of the SVM model is to design an optimal hyperplane which separates the examples of different classes for given training data points. The decision hyperplane is constructed by maximizing the distance of hyperplane and the nearest examples from different classes that are called support vectors 3.1.

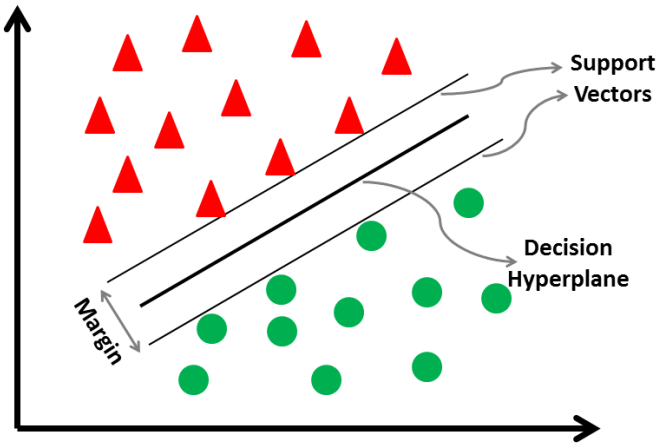


Figure 3.1: Support Vector Machine

In details, SVM method needs optimal solution of the problem below [33]:

$$\min_{w,b,\xi} \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \tag{3.1}$$

for (sample, class label) pairs of training set $(x_i, y_i)$ where $i = 1, ..., l$, $x_i \in R^n$ and $y \in 1, -1^l$,

subject to $y_i(w^T\phi(x_i) + b)$>=$1 - \xi_i$, $\xi_i$>=0.

Training vectors $x_i$ are mapped into a higher dimensional space by using the function $\phi$. SVM method produces a hyperplane with the maximal margin in this space. Lastly, C is the penalty parameter and $K(x_i, x_j) \equiv \phi(x_i)^T\phi(x_j)$ is the kernel function. The kernel function of SVM is too important in many cases because it is crucial for non-separable patterns. More detailed, if a pattern has separable data 3.2, SVM can find an optimal hyperplane between two classes easily.

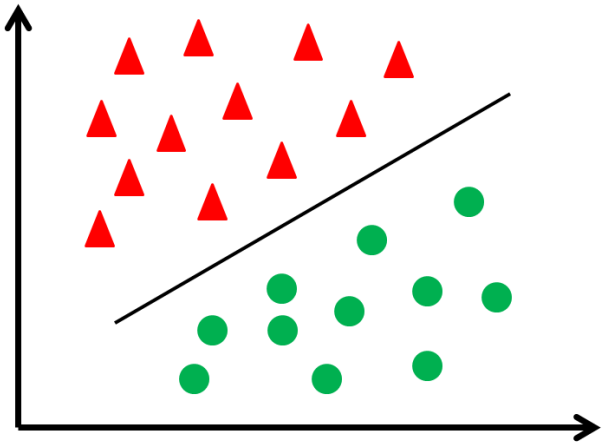

Figure 3.2: Separable Data Pattern

However, if the pattern is non-separable 3.3, SVM needs to map the current data into new space by transformations in order to make the current pattern separable. The transformation is processed with a predefined Kernel Function.

Mostly used kernel functions are Radial Basis Function (RBF), Linear, Sigmoid and Polynomial functions. We will write only Linear and RBF functions Linear Function Radial Basis Function(RBF) because we chose to apply them in this study. The reasons of chose of them are that RBF is most popular and commonly used function for SVM and Linear function is suggested for solving large-scale

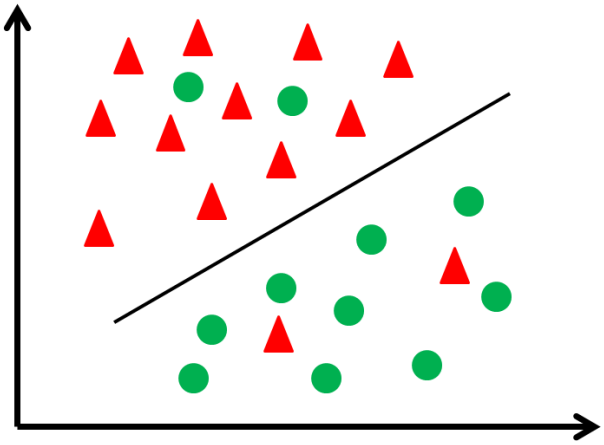

Figure 3.3: Non-separable Data Pattern

classification problems such as text classification [34].

$$K(x_i, x_j) = x_i^T x_j \qquad \text{(Linear Function)}$$

$$K(x_i, x_j) = exp(-\gamma \|x_i - x_j\|^2) \ , \gamma > 0 \qquad \text{(Radial Basis Function(RBF))}$$

## 3.2 Maximum Entropy

Maximum Entropy (ME), called MaxEnt in short, is a statistical classification modeling technique which was introduced by Berger et al.(1996) [35] in order to solve several natural language processing problems. Since then, the method has been used for lots of text classification studies [22] [23] [36] [37].

Training data includes relationship information between features and class types. The maximum entropy model uses this relationships in order to estimate probabilities. If training data is a text, this algorithm models conditional distribution

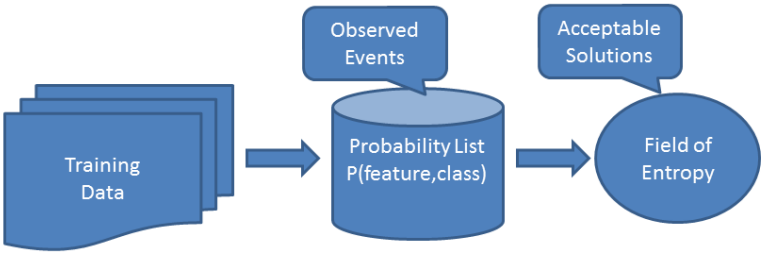


Figure 3.4: Maximum Entropy Model

of the words of texts in classes. Probabilistic distribution of a text classification model is computed as [35] :

$$p(c|d) = \frac{1}{Z(d)} exp(\sum_i \alpha_i f_i(d,c))) \tag{3.2}$$

where $Z(d)$ in 3.2 is a partition function which makes normalization. It is computed as:

$$Z(d) = \sum_c exp(\sum_i \alpha_i f_i(d,c))) \tag{3.3}$$

In equations 3.2 and 3.3, $c$ indicates class type, $d$ indicates document. The parameter $\alpha_i$ refers weight of feature and it must be learned by estimation. Various estimation algorithms could be used for this step such as Limited-Memory Variable Metric (L-BFGS) [38], Orthant Wise Limited-memory Quasi Newton (OWLQN) or Stochastic Gradient Descent (SGD) [39]. $f_i(d,c)$ indicates the impact of a feature $i$. The impact of the function could has binary or positive integer value. While binary value is used for occurrence of a word in text, integer value could give more information such as frequency of word. More precisely the function is formulated as [37]:

$$f_{(w,c')}(d,c) = \begin{cases} 0, & \text{if } c \neq c' \\ \frac{N(d,w)}{N(d)}, & \text{otherwise} \end{cases} \tag{3.4}$$

where N(d,w) in equation 3.4 is the density of word $w$ in document $d$ and $N(d)$ is the total density of words $d$.

### 3.3 Extreme Learning Machine

Extreme Learning Machines, called ELM in short, is a learning algorithm for single-hidden layer feed-forward neural networks(see Figure 3.5 for SLFN) which is introduced by Huang, Zhu and Siew in 2004 [40]. The main deficiency of feed-forward neural networks was the slowness problem because of slow gradient-based algorithms used in training step and tuning operation of all the parameters of the network iteratively. In order to handle this bottleneck, Huang et al. developed an algorithm that chooses the input weights randomly and decides analytically the output weights in order to obtain best generalization performance with extremely fast learning speed [40] [41] [30].

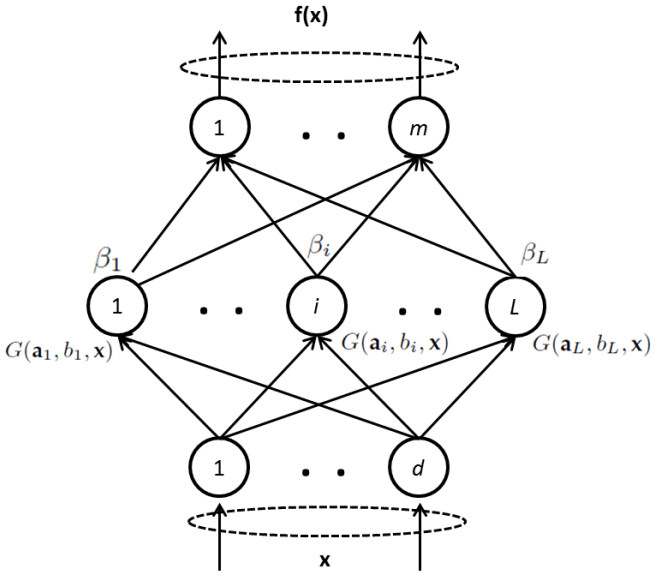


Figure 3.5: Single-hidden layer feed-forward network.

The output of SLFN having $L$ number of hidden nodes can be represented with the formula below;

$$f_L(x) = \sum_{i=1}^{L} \beta_i G(a_i, b_i, x), x \in R^n, a_i \in R^n \tag{3.5}$$

where learning parameters of hidden nodes are $a_i$ and $b_i$ and $\beta_i$ is the connnection weight between the $i$th hidden node and the output node. $G(a_i, b_i, x)$ is the

output of the hidden node with the input $x$. Generally, the additive hidden node based on activation function is $g(x) : R \to R$. $G(a_i, b_i, x)$ is given by

$$G(a_i, b_i, x) = g(a_i.x + b_i), b_i \in R, \tag{3.6}$$

where $a_i.x$ represents the inner product of $a_i \in R^n$ and $x \in R^n$ vectors. For the training samples $\{(x_i, t_i)\}_{i=1}^N \subset R^n \times R^m$, if output of the network is equal to the targets, we have

$$f_L(x_j) = \sum_{i=1}^{L} \beta_i g(a_i.x_j + b_i) = t_j, j = 1, ..., N \tag{3.7}$$

Equation 3.7 could be written as:

$$H\beta = T \tag{3.8}$$

$$\mathbf{H} = \begin{bmatrix} h(x_1) \\ . \\ . \\ . \\ h(x_N) \end{bmatrix} = \begin{bmatrix} G(a_1.b_1 + x_1) & . & . & . & G(a_L.b_L + x_1) \\ . & & . & & . \\ . & & . & & . \\ . & & . & & . \\ G(a_1.b_1 + x_N) & . & . & . & G(a_L.b_L + x_N) \end{bmatrix}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_1^T \\ . \\ . \\ . \\ \beta_L^T \end{bmatrix} \; and \; \mathbf{T} = \begin{bmatrix} t_1^T \\ . \\ . \\ . \\ t_L^T \end{bmatrix}$$

where $H$ is the hidden layer output matrix of the SLFN, the $i$th column of $H$ is the $j$th hidden node output that is relevant to the input $x_1, x_2, ..., x_N$. $h(x)$ is the hidden layer feature mapping. $h(x_i)$ is relevant to the the $i$th input $x_i$. It has been proved, if the activation function $G$ is infinitely differentiable in any

interval when the hidden layer parameters are randomly generated [41].

## 3.4 Machine Learning Algorithm Implementations for Our Problem

### 3.4.1 Support Vector Machines Settings

In order to understand SVM better and use the codes efficiently, the document *A Practical Guide to Support Vector Classification* [33] guided us. There are some important issues that have significant influence on SVM results.

First issue is scaling. If data is not scaled, greater numeric ranges may dominate smaller numeric ranges. Therefore, scaling ranges [-1,+1] and [0,1] are suggested. The range [0,1] was chosen on this study. Second issue is model selection. RBF Kernel is suggested as first choice because of its accuracy and popularity for SVM. Besides, Linear-SVM is also suggested for text classification with its speed and convenience for large-scaled data. Also, related works chose these two models which are formulated in Equation Linear Function and Equation Radial Basis Function(RBF).

Open source projects of these methods which are used in this study as base codes are LIBSVM [42] for RBF-SVM and LIBLINEAR [43] for Linear-SVM. The last issue is finding best parameters for these models. In order to handle this issue, we used cross-validation and grid-search solution. Basically, we divided our training set into five subsets and used four of them for training and one of them for testing with various pairs of parameters. After that, we chose $C$ for Linear-SVM and ($C$, $\gamma$) for RBF-SVM which belong to best accuracy on grid-searches. They were $32, 64, 64, 128, 32$ for Linear-SVM and $(128.0, 0.125), (8, 8), (8, 8), (32, 8), (8, 8)$ for RBF-SVM in order to model training set sizes 500, 2500, 5000, 25000, 50000 respectively. All values are powers of two because parsed values in grid are $2^-5, 2^-3, ..., 2^15$ for $C$ and $2^-15, 2^-13, ..., 2^3$ for $\gamma$. After parameters obtained, models were created for each algorithm and each training set. Then, feature vectors of each web page were predicted with created model and predictions were saved. Lastly, we obtained true positive,

false positive, true negative and false negative ratios by comparing actual results and predicted results of web pages. In addition, we kept the models in order to use them in future works and applications.

### 3.4.2 Maximum Entropy Settings

A maximum entropy classification library *MAXENT* which is implemented at Tsujii Laboratory in University of Tokyo [44] was arranged and used on this study. This library supports L1 and L2 regularizations. Besides, OWLQN, LBFGS and SGD algorithm for optimization. Firstly, we needed to update input sets because the values of the method could be integers in range [0,255]. Binary representation was not a problem but other types of inputs including floats were scaled again. As recommended, we executed L1 regularization with OWLQN and L2 regularization with LBFGS. Iteration count was selected as 300. Big iteration count caused more time consumption but it also increased accuracy. This program, firstly, saved all training data in a model. Then, it trained it and checked test set like SVM. At the result, models were saved in order to use again and also true positive, false positive, true negative and false negative ratios were obtained. Models of this algorithm include actively used features and their effects with a list such as;

### 3.4.3 Extreme Learning Machine Settings

Fundamental ELM codes were obtained from the web site of Nanyang Technological University [45]. Samples of this library suggests random data selection and more than one trial for each data set because each trial gives different result which is caused by random input weights on decision of output weights. Random selection of data samples was unnecessary for our problem because we had already created different sized sample sets. On the other hand, we got average performance for 50 trials of each sample set as recommended. The most important issue about this library is that it is implemented for small sizes of feature set. However, feature set size is very large in our problem so that matrix is very big and sparse on this study. Therefore, we arranged the open source code by

Table3.1: Small part of MaxEnt model

| Class Label | Feature ID | Lambda Weight of Feature |
|---|---|---|
| 0 | 10496 | 0.122787 |
| 1 | 1052 | -0.220663 |
| 0 | 1052 | 0.220663 |
| 1 | 1060 | -0.043466 |
| 0 | 1060 | 0.043466 |
| 0 | 10730 | 0.033599 |
| 0 | 1076 | 0.118291 |
| 1 | 11 | 0.004689 |
| 0 | 11 | -0.004689 |
| 0 | 1102 | 0.092983 |
| 1 | 1103 | -0.317303 |

using sparse matrices. Lastly, like the other algorithms executed, we obtained averages of true positive, false positive, true negative and false negative ratios in 50 trials results.

# CHAPTER 4

# EXPERIMENTAL SETUP

In this chapter, data preparation and experimental setup are described. Data preparation tasks and the experiments are carried out on an Intel Core I5-4570 3.20GHz computer with 8 GB RAM.

## 4.1 Experimental data sets

Two data sets are used in this study; Phistank(2016) [46] for malicious web sites and Alexa(2016) [47] for safe (benign) web sites because both of the data sets have proven themselves in earlier studies [13] [18] [21]. Phistank is a community site to submit, verify, track and share phishing data. Also, the community provides current malicious web site URLs list open sourced. On the other side, Alexa is an analytic tool and provides a list of top ranked 1 million web site URLs. We assume that pages of Alexa are benign because they are extremely popular and top ranked pages over the world. These data sets provide us 28848 malicious and 1 million benign URLs but we prefer to use top two hundred thousand of the benign page URLs in Table 4.1.

## 4.2 Data Preparation

Data preparation is a time consuming process because of nature of data mining. First issue is extracting meaningful data from the web pages. In order to handle this issue in small time and least error rate, we follow Comodo® Group

[48]keyword density extractor library's lead. Secondly, data preparing steps are prone to errors so that lots of small scripts and tests are prepared during preparation of database and files. Thirdly, because of the big data size described in subsections, some executions take more than a week. Although the preparation of the data is not the main focus of this study, we have seen that preparation phase of the data before processing with machine learning techniques can be time consuming. Parallel computing and big data techniques can be used in this area. Preparation tasks are explained detailed in sections below.

### 4.2.1 Crawling

In the crawling step, 28848 malicious and 200000 benign URLs are used at first in order to request HTML contents of web pages. However significant part of them are eliminated as shown on Table 4.1. 9% of the URLs are unreachable so they are eliminated. Then, language detection is executed by JLangDetect [49] on crawled HTML contents and 42% of them are also filtered because their language is not English or word count is smaller than five because if there is not sufficient content, both language detection and classification with content are meaningless issues. At the end of crawling step, we obtain 20799 malicious and 99974 benign English web page HTML contents save in text files as represented in Figure 4.1. Getting the HTML contents of web sites, saving them to text files, detection of their language and delay of unreachable URLs cause almost one week time wasting although 20 seconds timeout limit is used for unreachable URLs. Lastly a hundred thousand of the HTML contents of web pages are randomly selected in order to use them in this study.

Table4.1: Data Sets

| Counts | Malicious | Safe(Benign) |
|---|---|---|
| URL | 28848 | 200000 |
| Crawled Web Page | 26039 | 181665 |
| English Web Page | 20799 | 99974 |
| Used Web Page | 20000 | 80000 |


Figure 4.1: Crawling

### 4.2.2 Feature Extraction

We use words in HTML content of web pages in order to extract keywords as shown in Figure 4.2. In order to obtain correct feature set, firstly these contents are parsed and some conventional content process methods are used. Some of

the implemented processes could be summarized as below;

**Article Extraction:** Web pages include both valuable information and irrelevant texts. Article extraction helps obtaining only valuable information from a web page. In order to filter irrelevant texts; it removes navigation links, advertisements, menu items, selection items, videos, images etc.

**Removing Some Special Characters:** Removal of special characters, punctuations, apostrophes, words containing only one or two characters etc. increase clearness of text analyzing.

**Stemming:** Stemming process is summarized briefly as reducing derived or inflected words in order to obtain base form of the word. For instance, 'stems', 'stemmer', 'stemming' and 'stemmed' have same root 'stem' so each of them should be considered as root word.

**Stop Word Elimination:** Stop words generally refer to most common words in a language. Using them in text processing does not express a meaning. Some example stop words are 'but', 'are', 'some', 'the', 'who', 'and', 'etc'.

The second issue of feature extraction is scoring. Related studies generally use binary representation or TF-IDF because they are simple methods and they present satisfying results for text classification. However, web pages have more features than a regular text such as HTML tags so we obtain keyword frequencies of the HTML contents and compare with conventional methods. In other words, we prefer to use both *Binary Representation*, *TF-IDF* and *Keyword Density*.

#### 4.2.2.1 Binary Representation

This feature value type is only interested in a keyword occurs in a text or not. If the text $T$ contains the keyword $k$, value of feature $k$ in the feature vector of $T$ is 1. Otherwise, its value is 0. This method is easy and efficient in size and time because values are binary.


Figure 4.2: Feature Extraction

#### 4.2.2.2 Keyword Frequency and TF-IDF

Frequency of a word shows its importance in the text. However, some words create noisy like stop words. Therefore Term Frequency Inverse Document Frequency(TF-IDF) is mostly used approach in text mining rather than TF(Term Frequency). The formula of TF-IDF is multiplication of term frequency and inverse document frequency represented in Equation 4.1. This approach helps highlighting words that occur rarely in the all data set but frequently in the document [16].

$$tf_{t,d} * log(\frac{N}{n_t}) \tag{4.1}$$

where $tf_{t,d}$ represents term frequency in a document, N represents total document count which is 100000 in this study and $n_t$ represents the document count including the term $t$.
For an instance of keyword frequency, Wikipedia keyword extraction link `https://en.wikipedia.org/wiki/Keyword_extraction` is analyzed and 159 keywords are extracted AppendixB. Frequencies of the top ranked 10 keywords are listed in Table 4.2. According to the instance table, 'languag', 'document', 'method', 'process' and 'text' keywords are more relevant with this web page although frequency of 'term' is higher than their frequencies.

#### 4.2.2.3 Keyword Density

Keyword Frequency is successful and commonly used feature for document classification but HTML contents include more valuable properties. They are titles, meta keywords, headers and other HTML tags. It is not unquestionable issue that a word found in header or title is more valuable than another word in text. We analyze HTML content of a web page considering tags of HTML, then we score each keyword by considering its frequency and tags. This score is called as *density* in the rest of this study.
For an instance of keyword density, Wikipedia keyword extraction link `https://en.wikipedia.org/wiki/Keyword_extraction` is analyzed and 159 keywords

Table4.2: Top 10 Keyword Frequencies of a Wikipedia Web Page

| Rank | Keyword | TF | TF-IDF |
|---|---|---|---|
| 1 | keyword | 14 | 22.25620650 |
| 2 | extract | 9 | 18.44671631 |
| 3 | wikipedia | 8 | 18.63061725 |
| 4 | method | 6 | 8.48317784 |
| 5 | term | 5 | 2.20483814 |
| 6 | languag | 4 | 3.99531594 |
| 7 | text | 4 | 4.96495782 |
| 8 | assign | 3 | 5.82586130 |
| 9 | document | 3 | 2.82290098 |
| 10 | process | 3 | 3.16152781 |

are extracted AppendixB. Densities of the top ranked 10 keywords are listed in Table 4.3;

Table4.3: Top 10 Keyword Densities of a Wikipedia Web Page

| Rank | Keyword | Density Score | Density Ratio |
|---|---|---|---|
| 1 | keyword | 117.2 | 0.12752587 |
| 2 | extract | 111.5 | 0.12129783 |
| 3 | edit | 33.1 | 0.03600859 |
| 4 | assign | 27.3 | 0.02969893 |
| 5 | languag | 27.1 | 0.02948135 |
| 6 | term | 21.3 | 0.02317169 |
| 7 | text | 18.2 | 0.01979928 |
| 8 | method | 18.2 | 0.01979928 |
| 9 | search | 18.1 | 0.01969050 |
| 10 | process | 18.1 | 0.01969050 |

### 4.2.3 Feature Set Generation

In order to generate feature vectors, feature set should be defined so that all keywords in feature files and document frequencies of them are saved at first. The most time consuming step is creating this MYSQL table because of lots of select and update transactions on database especially after table growing so that this step took more than a week. After all keywords obtained, the table is simplified to keep only smaller and meaningful sets. At a result, we create five tables, one of them keeps all keywords which are found in keywords of one hundred thousand web pages and four of them are simplified versions of the all keywords table as shown in ER diagram Figure 4.3.

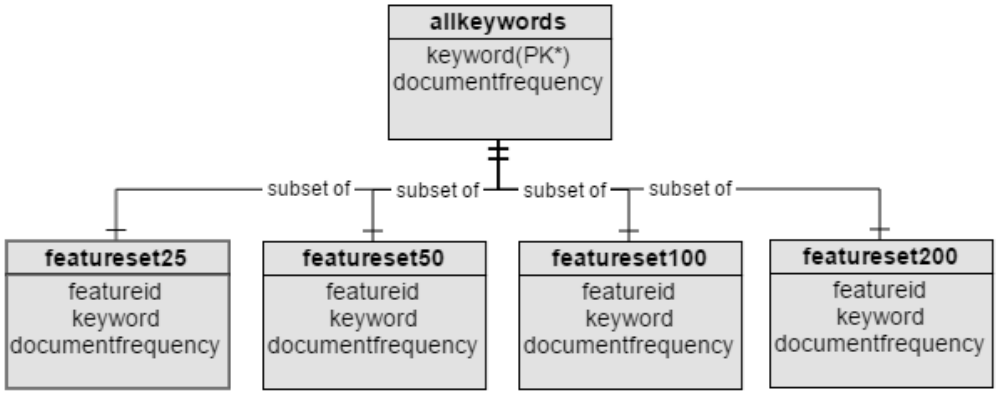


Figure 4.3: Entity Relationship Diagram

**allkeywords table:** The all keywords table is created with using all web pages' keywords as shown in Figure 4.4 so this table contains almost 800000 rows. Each row keeps a unique keyword which will be used as feature and document frequency of it. In order to decrease effort, this step could be handled by using ready common English keywords list instead of composing 'allkeywords' table but self constructed table contains lots of words which are not contained in English dictionary but exist in web. Sample row from the table below shows that *download* keyword has found in 17162 different documents as a keyword.
download, 17162

**featuresetN tables:** The whole rows in the allkeywords table should not

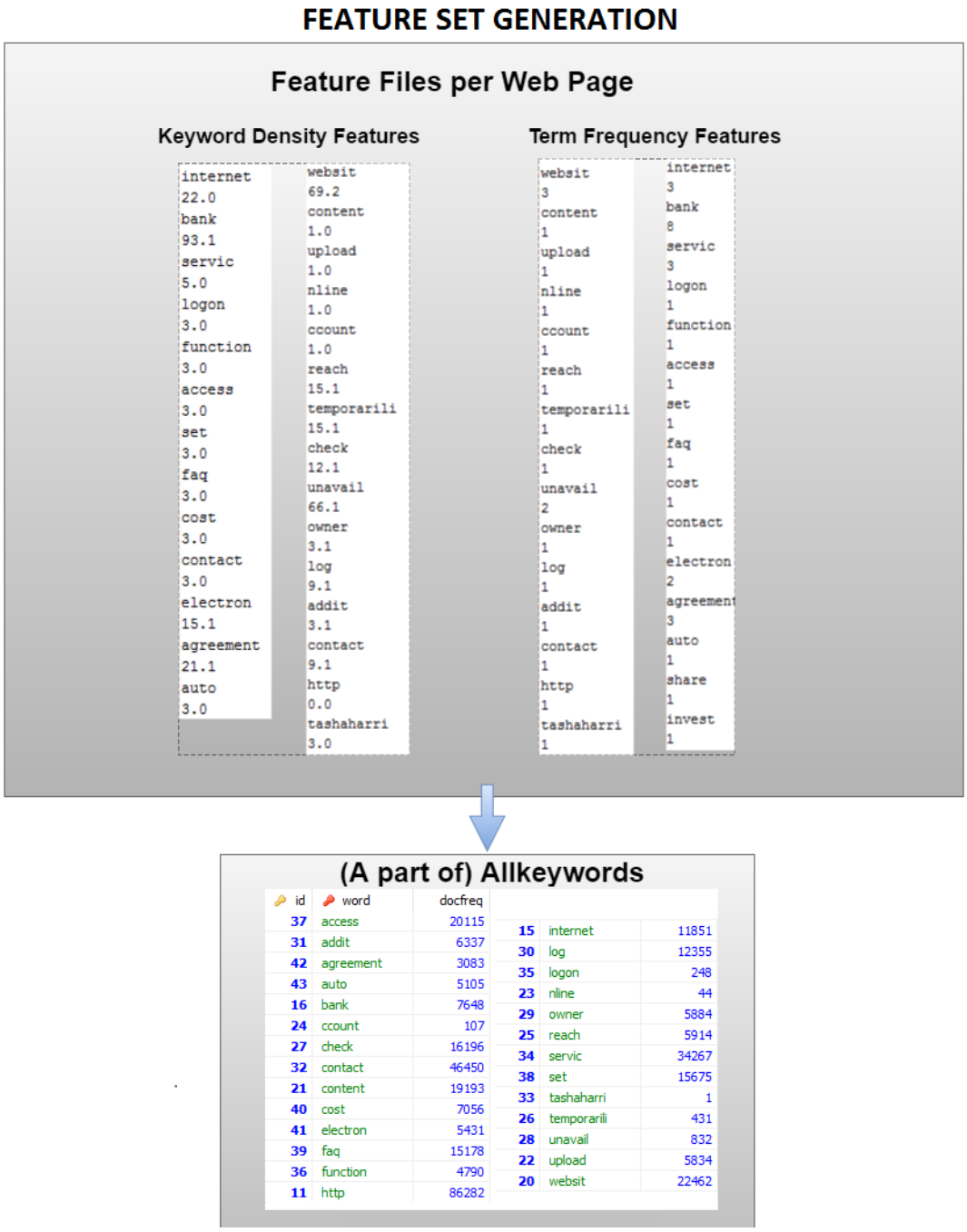


Figure 4.4: Feature Set

be considered as feature because the table includes words having spelling errors, space missing and language interfere such as 'installatiebedrijf', 'movingforward-trademarklogo', 'strategisch'. Luckily, frequencies of the problematic words give a clue about their validity. In order to handle this problem, feature sets are created by simplifying allkeywords table. In other words, the feature set tables

are subsets of allkeywords table. They are simplified tables containing the keywords whose document frequencies are bigger than a limit $n$. We create and use featureset25, featureset50, featureset100 and featureset200 tables which are containing 33148, 20638, 12988 and 8288 rows in sequence. Each row indicates a feature by keeping id, keyword and document frequency. A sample row of featureset200 table indicates that *download* keyword's document frequency is 17162 and its id is 862 in featureset200 table;
862, download, 17162

### 4.2.4 Conversion into Feature Vectors

Last step is expressing each web page as feature vector in order to use on ML methods. On this step, we create three vectors for each web page; existence based, TF-IDF based and keyword density based. All feature vectors contain maximum 100 features in order to reduce data size. Existence based feature vectors keep only distinct first 100 keywords in text because we do not have more meaningful data due to binary representation. TF-IDF based feature vectors contain top ranked 100 TF-IDF scores per web page. Similarly, KD based feature vectors contains ratios of top ranked 100 keyword density for each web page. Lastly, TF-IDF values scale to [0,1] but the scaling process of TF-IDF decreased successes of results so scaled versions are not used. Existence values do not need scaling because their value set is 0,1. Also, keyword density ratio values do not be scaled because these values are already in range [0,1]. Data format of training and testing data files is:
<class label> <feature id1>:<value1> <feature id2>:<value2> ...
.
.
.
where feature ids are in ascending order. A part of our keyword density file as a sample is shown in Figure 4.5.

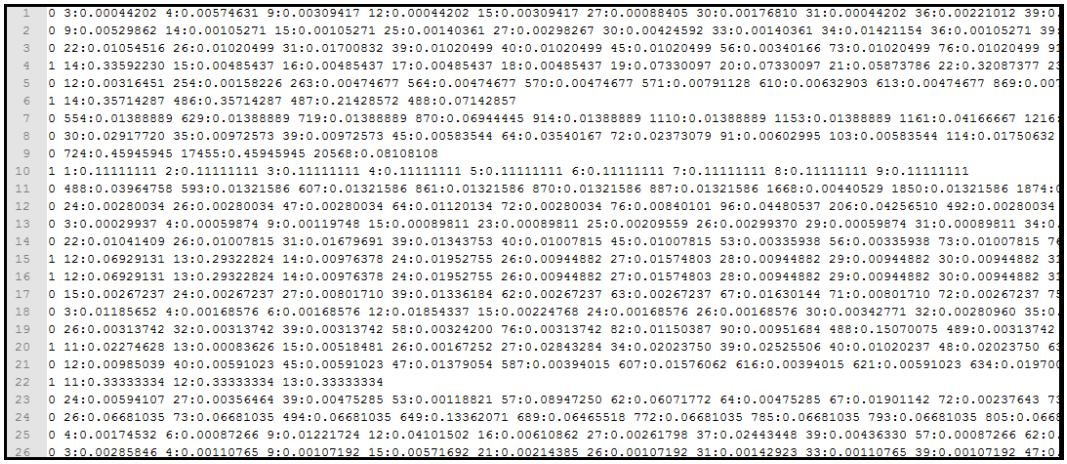


Figure 4.5: Feature File Sample

# CHAPTER 5

# PERFORMANCE EVALUATION OF THE PROPOSED METHODS

Supervised machine learning methods, SVM, ELM and ME, are executed for detection of malicious web pages. For this process, we have already extracted feature vectors for training and testing data sets in chapter 4. For the evaluation of these ML methods, after models are created with training sets, class labels of test sets are predicted with created model as shown in Figure 5.1 The performances of algorithms and other effects which may influence performance results are examined separately in sections below.

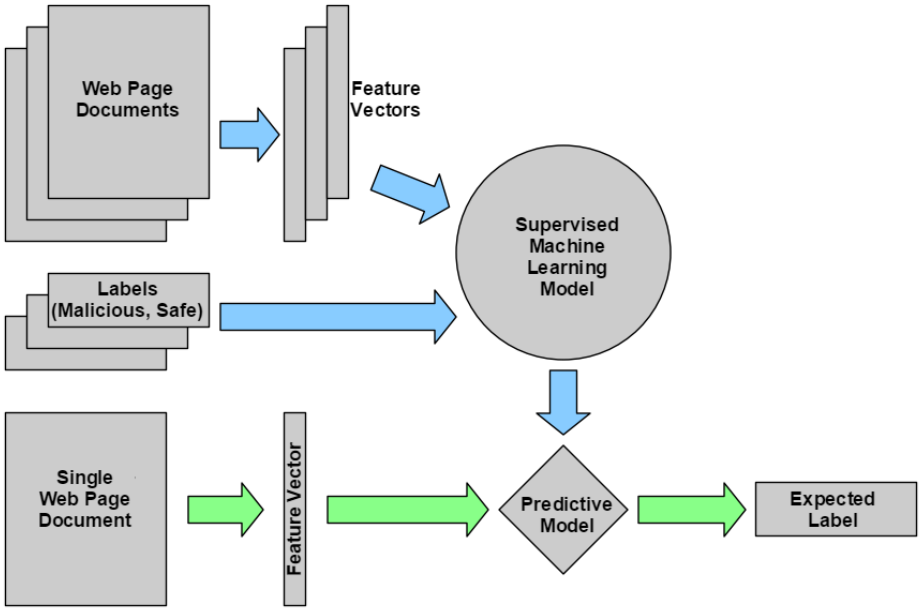


Figure 5.1: Supervised Machine Learning Model

## 5.1 The Effects of Data Set Size and Machine Learning Algorithms

Machine learning methods which are explained in chapter 3 are executed with different sizes of data sets. Data sets are composed randomly with 20000 malicious and 80000 safe web pages. Properties of data sets are described in Table 5.1

Table5.1: Properties of input sets

| **Total set size** | **1000** | **5000** | **10000** | **50000** | **100000** |
|---|---|---|---|---|---|
| **Fold count** | 100 | 20 | 10 | 2 | 1 |
| **Train set size** | 500 | 2500 | 5000 | 25000 | 50000 |
| **Test set size** | 500 | 2500 | 5000 | 25000 | 50000 |
| **Malicious count** | 100 | 500 | 1000 | 5000 | 10000 |
| **Safe count** | 400 | 2000 | 4000 | 20000 | 40000 |

Although researches show that the percentage of malicious web pages in the real world is roughly 0.1% [21], this percentage is not proper to generate a predictive model with ML algorithms. Therefore, we use 20% percentage for malicious web pages. In other words 1:4 ratio of malicious to safe web pages are used. Half of the web pages are used for training and other half for testing. Also, the same ratio of malicious web pages to safe web pages is used in test and train sets. Keywords density percentages in web pages are used as features so that feature values are between [0.0, 1.0] and their sum is 1 for each web page feature vector. Feature vector size is limited with maximum weighted 100 features for each web page. Besides, used feature set, featureset25, includes the features whose document counts are bigger than 25.
Averages of accuracies are listed in the Table 5.2 where accuracy is defined as successfully labeled web pages divided by all web pages.

Results are obtained by getting average accuracies of n different parts of $100000/n$ data sets. For instance, different 10 tests are done with the data including 10000 sample and averages are listed in the Table 5.2. Results are obtained by getting average accuracies of n different parts of $100000/n$ data sets. For instance, different 10 tests are done with the data including 10000 sample and averages are listed on the Table 5.2. We compare the algorithms with their success with

Table5.2: The Effects of Data Set Size and ML Algorithms on Accuracy(%)

| **Number of web pages:** | **1000** | **5000** | **10000** | **50000** | **100000** |
|---|---|---|---|---|---|
| **RBF-SVM** | 94.60 | 97.24 | 97.44 | **98.24** | 98.01 |
| **Linear-SVM** | 94.80 | 97.36 | 97.26 | **97.75** | 97.55 |
| **ME-L1** | 94.80 | 96.56 | 96.72 | **97.33** | 96.94 |
| **ME-L2** | 95.00 | 96.72 | 97.20 | **97.28** | 96.81 |
| **ELM** | 87.29 | 88.26 | 88.13 | 88.74 | 88.05 |

50000 sample because most successful results are obtained with this sample size. Their accuracies in descending order are RBF-SVM (98.24), Linear-SVM (97.75), MaxEnt-L1 (97.33), MaxEnt-L2 (97.28) and ELM (88.74).

Another success concern is a confusion matrix which is used for describing the performances of implemented classification models. It contains not only successfully detected malicious web page percentage but also the percentages of mislabeled web pages. It is important for our problem because false positive rate is also important issue on this study. In Table 5.3, the confusion matrix contains four sections. True Positive Rate(TPR) section indicates the percentage of successfully detected malicious web pages. True Negative Rate(TNR) section indicates the percentage of successfully detected safe web pages. False Positive Rate(FPR) represents the percentage of safe web pages labeled as malicious incorrectly. False Negative Rate(FNR) represents the percentage of malicious web pages labeled as safe incorrectly.

The algorithm success could be determined with high true positive and low false positive ratios. Their TPR ratios, also called recalls, in descending order is MaxEnt-L1 (94.68), MaxEnt-L2 (94.36), Linear-SVM (94.08), RBF-SVM (93.22) and ELM (48.93). Also, MaxEnt-L1 obtains active features on training step in order to speed up test. By examining these active words, we find the chance of listing valuable malicious and safe web page words in AppendixC and AppendixD On the other side, their FPR ratios in ascending order is RBF-

Table5.3: Confusion Matrices for 50000 web pages

| Prediction | | |
|---|---|---|
| | **Malicious** | **Safe** |
| **Actual Value M** | True Positive | False Negative |
| **Actual Value S** | False Positive | True Negative |

| RBF-SVM - Prediction | | |
|---|---|---|
| | **M** | **S** |
| **Actual Value M** | 93.22 | 6.78 |
| **Actual Value S** | 0.50 | **99.50** |

| Linear-SVM - Prediction | | |
|---|---|---|
| | **M** | **S** |
| **Actual Value M** | 94.08 | 5.92 |
| **Actual Value S** | 1.33 | 98.67 |

| L1 Reg ME - Prediction | | |
|---|---|---|
| | **M** | **S** |
| **Actual Value M** | **94.68** | 5.32 |
| **Actual Value S** | 2.01 | 97.99 |

| L2 Reg ME - Prediction | | |
|---|---|---|
| | **M** | **S** |
| **Actual Value M** | 94.36 | 5.64 |
| **Actual Value S** | 1.99 | 98.01 |

| ELM - Prediction | | |
|---|---|---|
| | **M** | **S** |
| **Actual Value M** | 48.93 | 51.07 |
| **Actual Value S** | 1.31 | 98.69 |

SVM (0.50), ELM (1.31), Linear-SVM (1.33), MaxEnt-L2 (1.99) and MaxEnt-L1. Even FPR ratio of ELM low, it does not show success of this method due to the very low TPR ratio of it. All in all, the successes of other methods are very close.

## 5.2 The Effect of Feature Value Type

The novelty of this study is feature value type because other studies generally use binary representation or TF-IDF. In addition to these conventional methods, we also try to represent results of keyword densities. These three methods are explained in details in subsubsection 4.2.2.1, subsubsection 4.2.2.2 and subsubsection 4.2.2.3.

In this section, we will compare their successes. These experiments are only conducted with 10 fold data sets including 10000 samples which contains 5000 training and 5000 test samples. Because of the successes of algorithms previous section, these tests are only executed with Linear-SVM and MaxEnt-L1.

Feature vector sizes are limited with 100 features. Selection of a hundred features and scaling are explained in chapter 4. According to the Table 5.4 and Table 5.5, Keyword Density is the best option as accuracy, result and f-measure but gaps are very small with binary representation.

Table5.4: The Effects of Feature Value Types on Accuracy

| % | **Accuracy** | **TPR** | **TNR** | **FPR** | **FNR** |
|---|---|---|---|---|---|
| **Linear-SVM** | | | | | |
| **Binary Representation** | 96.98 | 91.70 | 98.30 | 1.70 | 8.30 |
| **TF-IDF** | 96.92 | **92.20** | 98.10 | 1.90 | 7.80 |
| **Keyword Density** | **97.26** | 92.00 | **98.58** | 1.42 | 2.74 |
| **MaxEnt-L1** | | | | | |
| **Binary Representation** | 96.26 | 91.10 | 97.55 | 2.45 | 3.74 |
| **TF-IDF** | 94.64 | 91.20 | 95.50 | 4.50 | 8.80 |
| **Keyword Density** | **96.72** | **92.40** | **97.80** | 2.20 | 3.28 |

## 5.3 The Effect of Feature Set Size

We save extracted keywords from one hundred thousand web pages in order to use them as feature set. However, we need to put a lower limit to their

Table5.5: The Effects of Feature Value Types about Statistical Analysis

| % | Recall | Precision | F-Measure |
|---|---|---|---|
| **Linear-SVM** | | | |
| **Binary Representation** | 91.70 | 93.10 | 92.39 |
| **TF-IDF** | **92.20** | 92.38 | 92.29 |
| **Keyword Density** | 92.00 | **94.19** | **93.08** |
| **MaxEnt-L1** | | | |
| **Binary Representation** | 91.10 | 90.29 | 90.69 |
| **TF-IDF** | 91.20 | 83.52 | 87.19 |
| **Keyword Density** | **92.40** | **91.30** | **91.85** |

document frequencies due to the problematic words. In order to check efficiency or inefficiency of this limit, different number of document frequency limits are put into the keywords table and results are compared.

These experiments are only conducted by 10 fold data sets including 10000 sample which contains 5000 training and 5000 test samples. Feature vector sizes are limited with 100 features for each web page again. Because of the successes of algorithms in previous section, these tests are only executed with Linear-SVM and MaxEnt-L1 Regularization. All keywords table (FeatureSet0) is reduced to 4 tables. As shown in Table 5.6, row count of these tables are decreased from almost 800000 to almost 8000. Also, average number of different features in vectors of training set, that is used for modeling, is decreased by five times.

Table5.6: Feature Sets

| | Table Row Count | Number of Features |
|---|---|---|
| **FeatureSet0** | 789946 | 41230 |
| **FeatureSet25** | 33148 | 21980 |
| **FeatureSet50** | 20638 | 17169 |
| **FeatureSet100** | 12988 | 12257 |
| **FeatureSet200** | 8288 | 8173 |

According to the results of Table 5.7 and Table 5.8, there is no difference on

accuracy of methods although feature set size significantly decreases.

Table5.7: The Effects of Feature Set Size on Accuracy(%)

| | **Accuracy** | **TPR** | **TNR** | **FPR** | **FNR** |
|---|---|---|---|---|---|
| **Linear-SVM** | | | | | |
| **FeatureSet0** | 97.34 | 89.50 | 99.30 | 0.70 | 10.50 |
| **FeatureSet25** | 97.26 | 92.00 | 98.58 | 1.42 | 8.00 |
| **FeatureSet50** | 97.30 | 89.80 | 99.18 | 0.82 | 10.20 |
| **FeatureSet100** | 97.34 | 89.90 | 99.20 | 0.80 | 10.10 |
| **FeatureSet200** | 97.00 | 88.10 | 99.23 | 0.77 | 11.90 |
| **MaxEnt-L1** | | | | | |
| **FeatureSet0** | 96.74 | 90.70 | 98.25 | 1.75 | 9.30 |
| **FeatureSet25** | 96.72 | 92.40 | 97.80 | 2.20 | 7.60 |
| **FeatureSet50** | 96.60 | 90.10 | 98.23 | 1.77 | 9.90 |
| **FeatureSet100** | 96.74 | 91.50 | 98.05 | 1.95 | 8.50 |
| **FeatureSet200** | 96.64 | 90.20 | 98.25 | 1.75 | 9.80 |

Table5.8: The Effects of Feature Set Size on Statistical Analysis

| | **Recall(%)** | **Precision(%)** | **F-measure(%)** |
|---|---|---|---|
| **Linear-SVM** | | | |
| **FeatureSet0** | 89.50 | 96.97 | 93.09 |
| **FeatureSet25** | 92.00 | 94.19 | 93.08 |
| **FeatureSet50** | 89.80 | 96.48 | 93.02 |
| **FeatureSet100** | 89.90 | 96.56 | 93.11 |
| **FeatureSet200** | 88.10 | 96.62 | 92.16 |
| **MaxEnt-L1** | | | |
| **FeatureSet0** | 90.70 | 92.84 | 91.76 |
| **FeatureSet25** | 92.40 | 91.30 | 91.85 |
| **FeatureSet50** | 90.10 | 92.71 | 91.39 |
| **FeatureSet100** | 91.50 | 92.15 | 91.82 |
| **FeatureSet200** | 90.20 | 92.80 | 91.48 |

Equality on results despite of relatively feature count could be explained by

active feature count. Actually, even fewer features are used for classification actively. For example, due to the active feature lists of MaxEnt-L1 models approximately 1000 features are only used for classification according to the Table 5.9 although training sets have up to 41230 different features.

Table5.9: Active Feature Count

| | **Number of Features** | **Active Feature Count** |
|---|---|---|
| **FeatureSet0** | 41230 | 1008 |
| **FeatureSet25** | 21980 | 1048 |
| **FeatureSet50** | 17169 | 1060 |
| **FeatureSet100** | 12257 | 1059 |
| **FeatureSet200** | 8173 | 1071 |

## 5.4 The Running Times of Algorithms

In this section, we compare running times of algorithms by representing elapse time with the set used in section 5.1. The set has 5000 training and 5000 test feature vectors.

Table5.10: Running Time of ML Algorithms in seconds

| | **ME-L1** | **ME-L2** | **Linear-SVM** | **RBF-SVM** | **ELM** |
|---|---|---|---|---|---|
| **Train Time** | 12.94 | 8.21 | 0.02 | 5.13 | 0.09 |
| **Test Time** | 0.54 | 3.49 | 0.03 | 4.01 | 0.04 |

As shown in Table 5.10, best algorithms considering running time are Linear-SVM and ELM as expected because the power of these algorithms are low time consumptions. MaxEnt with L1 seems poor in training but the important part is testing time. Therefore, it is also successful. Lastly, RBF-SVM algorithm is the worst one because it includes much more calculation.

# CHAPTER 6

# CONCLUSION

In this study, we have proposed a malicious web page detection methodology which uses HTML content of web pages with three supervised ML methods. The implemented classification methods are Support Vector Machine, Maximum Entropy, and Extreme Learning Machines.

During the experiments, 20000 malicious and 80000 safe web pages are crawled and analyzed. The extracted features are the keywords of web pages' HTML contents. These web pages are split into same size training and test subsets. Besides, both training and test sets have same malicious:safe web page ratio; 1:4.

As the result of the experiments, ELM shows the worst results evidently not only for accuracy but also for true positive rate. On the other side, SVM and ME show similar success rates. While RBF-SVM gets the best accuracy with 98.24%, Linear-SVM, MaxEnt-L1 and MaxEnt-L2 methods closely follow RBF-SVM with accuracies more than 97.28%. Therefore, ranking these four methods is not healthy. Similarly, MaxEnt-L1 get the best (true positive, true negative) pair with (94.68%, 97.99%), MaxEnt-L2, Linear-SVM and RBF-SVM methods very closely follow it. Among the four methods, time is also essential issue. Considering as time, Linear-SVM, ELM and MaxEnt-L1 have similar and best methods. To summarize the results of ML methods, Linear-SVM or MaxEnt-L1 showed satisfactory result because of their high accuracies, low false positive rates and low running time. In a comprehensive manner, we also showed that these two methods are efficient with very much feature count and sample size

so they are suitable for other studies including problems about size issues.

On the other side, success of this study can be displayed by comparing with similar studies. Most similar and recent study [18] gets (true positive, false positive) pairs in the following order RBF-SVM(97.8%, 55.1%), Linear-SVM(92.4%, 83.2%), Naïve Bayes(76.4%, 87.4%) and K-Nearest Neighbor(9.9%, 94.8%). Although, study of Kazemian and Ahmed uses additional features, URL, page link and visual, results of our study are significantly better than it. The probable reason of this difference is preprocessing the content and feature extraction issues. Firstly, we clear up the keywords by using lots of conventional methods such as stemming, article extraction, stemming character elimination etc. Secondly, our thesis contributed a novel keyword extractor. While the conventional feature value types are binary representation and TF-IDF on text classification studies, we also extract keyword densities of web pages and used them as features. Feature value types do not affect very much hence keyword density shows the best results and TF-IDF shows the worst. Therefore, we showed that preprocessing of web pages should not be considered as only text processing.

Lastly, feature set elimination is tested because total feature count reaches almost eight hundred thousands keywords and used feature count in sample set exceeds forty thousands keywords. Elimination of the keywords having least document frequency decrease feature count. Accuracy or true positive rate does not change although total feature count decreases by magnitude of 95 and used feature count in sample set decreases by magnitude four.

There are some possibilities that could be thought as future work. (1) A hybrid feature set system can be created including not only keyword densities but also JavaScript functions, ActiveX objects, DNS-Server relationships and URL features. However, each feature should be analyzed separately so there is an additional work for weights of their effect on final decision. (2) Preprocessing and running times may be decreased with parallel processing. However, this work requires reimplementing of scripts and ML methods. (3) Feature selection can be added to this study because we showed that even a thousand of keywords are

sufficient for modeling. Decrease of feature set size will also probably decrease time and memory consumption.

# REFERENCES


[1] Buzzfeed. https://www.buzzfeed.com/denverpolicedeptartment/buying-selling-safely-craigslist-safety-tip-h0ue. Accessed: 2010-07-04.

[2] vimeo. https://vimeo.com/user36717388. Accessed: 2010-07-04.

[3] Devolutions. https://password.devolutions.net/Home/Crack-Serialz-Warez. Accessed: 2010-07-04.

[4] Texas business leads. http://texasbusinessleads.tumblr.com/. Accessed: 2010-07-04.

[5] iconfinder. https://www.iconfinder.com/icons/333597/account_admin_crime_hacker_log_in_open_profile_protection_secure_secured_security_unlock_user_icon. Accessed: 2010-07-04.

[6] tumblr. http://fyeahelise.tumblr.com/post/115446545183. Accessed: 2010-07-04.

[7] toy-projects. http://www.toy-projects.be/. Accessed: 2010-07-04.

[8] Iconfinder. https://www.iconfinder.com/icons/307577/code_document_file_html_programming_script_icon. Accessed: 2010-07-04.

[9] Pinterest • the world's catalog of ideas. https://www.pinterest.com/pin/215469163399119945/. Accessed: 2010-07-04.

[10] International Telecommunication Union. Statistics. http://www.itu.int/en/ITU-D/Statistics/Documents/statistics/2015/ITU_Key_2005-2015_ICT_data.xls, 2015.

[11] Ram Basnet, Srinivas Mukkamala, and Andrew H Sung. Detection of phishing attacks: A machine learning approach. In *Soft Computing Applications in Industry*, pages 373–383. Springer, 2008.

[12] Gary Wassermann and Zhendong Su. Static detection of cross-site scripting vulnerabilities. In *2008 ACM/IEEE 30th International Conference on Software Engineering*, pages 171–180. IEEE, 2008.

[13] Abubakr Sirageldin, Baharum B Baharudin, and Low Tang Jung. Malicious web page detection: A machine learning approach. In *Advances in Computer Science and its Applications*, pages 217–224. Springer, 2014.

[14] Niels Provos, Dean McNamee, Panayiotis Mavrommatis, Ke Wang, Nagendra Modadugu, et al. The ghost in the browser: Analysis of web-based malware. *HotBots*, 7:4–4, 2007.

[15] Pawan Prakash, Manish Kumar, Ramana Rao Kompella, and Minaxi Gupta. Phishnet: predictive blacklisting to detect phishing attacks. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–5. IEEE, 2010.

[16] Sushma Nagesh Bannur, Lawrence K Saul, and Stefan Savage. Judging a site by its content: learning the textual, structural, and visual features of malicious web pages. In *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence*, pages 1–10. ACM, 2011.

[17] Mihai Christodorescu and Somesh Jha. Testing malware detectors. *ACM SIGSOFT Software Engineering Notes*, 29(4):34–44, 2004.

[18] HB Kazemian and S Ahmed. Comparisons of machine learning techniques for detecting malicious webpages. *Expert Systems with Applications*, 42(3):1166–1177, 2015.

[19] Juan Chen and Chuanxiong Guo. Online detection and prevention of phishing attacks. In *Communications and Networking in China, 2006. ChinaCom'06. First International Conference on*, pages 1–7. IEEE, 2006.

[20] Alexander Moshchuk, Tanya Bragin, Damien Deville, Steven D Gribble, and Henry M Levy. Spyproxy: Execution-based detection of malicious web content. In *USENIX Security*, 2007.

[21] Luca Invernizzi, Paolo Milani Comparetti, Stefano Benvenuti, Christopher Kruegel, Marco Cova, and Giovanni Vigna. Evilseed: A guided approach to finding malicious web pages. In *Security and Privacy (SP), 2012 IEEE Symposium on*, pages 428–442. IEEE, 2012.

[22] Kamal Nigam, John Lafferty, and Andrew McCallum. Using maximum entropy for text classification. In *IJCAI-99 workshop on machine learning for information filtering*, volume 1, pages 61–67, 1999.

[23] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.

[24] Michael Chau and Hsinchun Chen. A machine learning approach to web page filtering using content and structure analysis. *Decision Support Systems*, 44(2):482–494, 2008.

[25] Christian Seifert, Ian Welch, Peter Komisarczuk, Chiraag Uday Aval, and Barbara Endicott-Popovsky. Identification of malicious web pages through

analysis of underlying dns and web server relationships. In *LCN*, pages 935–941. Citeseer, 2008.

[26] Christian Seifert, Ian Welch, and Peter Komisarczuk. Identification of malicious web pages with static heuristics. In *Telecommunication Networks and Applications Conference, 2008. ATNAC 2008. Australasian*, pages 91–96. IEEE, 2008.

[27] Yung-Tsung Hou, Yimeng Chang, Tsuhan Chen, Chi-Sung Laih, and Chia-Mei Chen. Malicious web content detection by machine learning. *Expert Systems with Applications*, 37(1):55–60, 2010.

[28] Davide Canali, Marco Cova, Giovanni Vigna, and Christopher Kruegel. Prophiler: a fast filter for the large-scale detection of malicious web pages. In *Proceedings of the 20th international conference on World wide web*, pages 197–206. ACM, 2011.

[29] Ahmed Abbasi, Fatemeh Zahedi, Siddharth Kaza, et al. Detecting fake medical web sites using recursive trust labeling. *ACM Transactions on Information Systems (TOIS)*, 30(4):22, 2012.

[30] Guang-Bin Huang, Dian Hui Wang, and Yuan Lan. Extreme learning machines: a survey. *International Journal of Machine Learning and Cybernetics*, 2(2):107–122, 2011.

[31] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM, 1992.

[32] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[33] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. A practical guide to support vector classification. 2003.

[34] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *Journal of machine learning research*, 9(Aug):1871–1874, 2008.

[35] Adam L Berger, Vincent J Della Pietra, and Stephen A Della Pietra. A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71, 1996.

[36] Hai Leong Chieu and Hwee Tou Ng. A maximum entropy approach to information extraction from semi-structured and free text. *AAAI/IAAI*, 2002:786–791, 2002.

[37] Alaa El-Halees. Arabic text classification using maximum entropy. *The Islamic University Journal (Series of Natural Studies and Engineering)*, 15(1):157–167, 2007.

[38] Jorge Nocedal. Updating quasi-newton matrices with limited storage. *Mathematics of computation*, 35(151):773–782, 1980.

[39] Yoshimasa Tsuruoka, Jun'ichi Tsujii, and Sophia Ananiadou. Stochastic gradient descent training for l1-regularized log-linear models with cumulative penalty. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 477–485. Association for Computational Linguistics, 2009.

[40] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: a new learning scheme of feedforward neural networks. In *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, volume 2, pages 985–990. IEEE, 2004.

[41] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: theory and applications. *Neurocomputing*, 70(1):489–501, 2006.

[42] Chih-Chung Chang and Chih-Jen Lin. Libsvm – a library for support vector machines. `https://www.csie.ntu.edu.tw/~cjlin/libsvm/`, 2015.

[43] Machine Learning Group at National Taiwan University. Liblinear – a library for large linear classification. `https://www.csie.ntu.edu.tw/~cjlin/liblinear/`, 2015.

[44] Yoshimasa Tsuruoka. A simple c++ library for maximum entropy classification v3.0. *Software available at http://www.nactem.ac.uk/tsuruoka/maxent/*, 2006.

[45] Zhu Qin-Yu and Huang Guang-Bin. Basic elm algorithms. `http://www.ntu.edu.sg/home/egbhuang/elm_codes.html`, 2004.

[46] PhishTank. Join the fight against phishing. `https://www.phishtank.com/developer_info.php`, 2016.

[47] Alexa. Alexa top sites. `http://s3.amazonaws.com/alexa-static/top-1m.csv.zip`, 2016.

[48] Comodo Group. Creating trust online. `https://www.comodo.com/`, 2016.

[49] Cédric Champeau. Jlangdetect. `https://github.com/melix/jlangdetect`, 2014.

# APPENDIX A

## MYSQL DATABASE QUERIES

Database queries below are used for saving and using feature sets on MySQL Database. Properties of these tables are explain in subsection 4.2.3.

```
CREATE TABLE `allkeywords` (
  `keyword` VARCHAR(255) NOT NULL,
  `existencecount` INT(11) NOT NULL DEFAULT '1',
  PRIMARY KEY (`keyword`)
);
```

```
CREATE TABLE `featureset25` (
  `featureid` INT(11) NOT NULL AUTO_INCREMENT,
  `keyword` VARCHAR(255) NULL DEFAULT NULL,
  `existencecount` INT(11) NULL DEFAULT '1',
  PRIMARY KEY (`newid`),
  INDEX `word` (`word`)
);
```

# APPENDIX B

## WEB PAGE ANALYSIS

In this study, Keyword Density extractor library which is designed by Comodo® Group [48] is used in order to analyze HTML contents of web pages. In this analysis, each word in the web page passes some conventional text analysis processes. After that, web page is defined as a keyword list which has also term frequency of each keyword and keyword density score in order to calculate their weights for feature vectors. Table below shows a keyword list of a sample web page, `https://en.wikipedia.org/wiki/Keyword_extraction`.

TableB.1: Keywords of a Wikipedia Web Page

| Keyword | Term Frequency | Keyword Density Score |
|---|---|---|
| keyword | 14 | 117.22497216032181 |
| extract | 9 | 111.5 |
| wikipedia | 8 | 15.0 |
| free | 2 | 18.0 |
| encyclopedia | 2 | 18.0 |
| assign | 3 | 27.3 |
| edit | 3 | 33.1 |
| refer | 1 | 4.0 |
| navig | 3 | 30.1 |
| menu | 1 | 4.0 |
| jump | 1 | 3.1 |
| search | 2 | 18.1 |
| task | 2 | 6.2 |

| | | |
|---|---|---|
| automat | 2 | 6.2 |
| identif | 1 | 3.1 |
| term | 5 | 21.3 |
| subject | 1 | 3.1 |
| document | 3 | 9.3 |
| kei | 3 | 9.3 |
| phrase | 1 | 3.1 |
| segment | 1 | 3.1 |
| terminolog | 2 | 6.2 |
| defin | 1 | 3.1 |
| repres | 1 | 3.1 |
| relev | 1 | 3.1 |
| contain | 1 | 3.1 |
| function | 1 | 3.1 |
| character | 1 | 3.1 |
| topic | 1 | 3.1 |
| discuss | 1 | 3.1 |
| problem | 1 | 3.1 |
| text | 4 | 18.2 |
| mine | 1 | 9.1 |
| retriev | 2 | 12.1 |
| natur | 3 | 18.1 |
| languag | 4 | 27.1 |
| process | 3 | 18.1 |
| method | 6 | 18.2 |
| roughli | 1 | 3.1 |
| divid | 2 | 6.1 |
| chosen | 2 | 6.2 |
| control | 1 | 3.1 |
| vocabulari | 1 | 3.1 |
| taxonomi | 1 | 3.1 |

| | | |
|---|---|---|
| explicitli | 1 | 3.1 |
| mention | 1 | 3.1 |
| origin | 1 | 3.1 |
| supervis | 2 | 6.1 |
| semi | 1 | 1.0 |
| unsupervis | 2 | 2.0 |
| simpl | 1 | 1.0 |
| statist | 1 | 1.0 |
| linguist | 1 | 1.0 |
| graph | 2 | 2.0 |
| base | 3 | 3.0 |
| beliga | 2 | 2.0 |
| slobodan | 2 | 2.0 |
| ana | 2 | 2.0 |
| sanda | 2 | 2.0 |
| overview | 1 | 1.0 |
| approach | 1 | 1.0 |
| journal | 1 | 1.0 |
| organiz | 1 | 1.0 |
| scienc | 1 | 1.0 |
| rada | 1 | 1.0 |
| mihalcea | 1 | 1.0 |
| paul | 1 | 1.0 |
| tarau | 1 | 1.0 |
| textrank | 1 | 1.0 |
| bring | 1 | 1.0 |
| order | 1 | 1.0 |
| proceed | 1 | 1.0 |
| confer | 1 | 1.0 |
| empir | 1 | 1.0 |
| emnlp | 1 | 1.0 |

| | | |
|---|---|---|
| barcelona | 1 | 1.0 |
| spain | 1 | 1.0 |
| juli | 1 | 1.0 |
| select | 1 | 1.0 |
| croatian | 1 | 1.0 |
| new | 1 | 1.0 |
| surfac | 1 | 1.0 |
| deep | 1 | 1.0 |
| social | 1 | 1.0 |
| web | 1 | 1.0 |
| sdsw | 1 | 1.0 |
| itali | 1 | 1.0 |
| ceur | 1 | 1.0 |
| proc | 1 | 1.0 |
| categori | 1 | 3.0 |
| person | 1 | 3.0 |
| tool | 2 | 5.0 |
| log | 2 | 4.0 |
| talk | 2 | 5.0 |
| contribut | 1 | 3.0 |
| creat | 2 | 5.0 |
| account | 1 | 3.0 |
| namespac | 1 | 3.0 |
| articl | 2 | 5.0 |
| variant | 1 | 3.0 |
| view | 3 | 8.0 |
| read | 1 | 3.0 |
| histori | 1 | 3.0 |
| main | 1 | 3.0 |
| content | 2 | 5.0 |
| featur | 1 | 3.0 |

| | | |
|---|---|---|
| current | 1 | 3.0 |
| event | 1 | 3.0 |
| random | 1 | 3.0 |
| donat | 1 | 3.0 |
| store | 1 | 3.0 |
| interact | 1 | 3.0 |
| commun | 1 | 3.0 |
| portal | 1 | 3.0 |
| contact | 2 | 5.0 |
| link | 3 | 7.0 |
| upload | 1 | 3.0 |
| file | 1 | 3.0 |
| special | 1 | 3.0 |
| perman | 1 | 3.0 |
| cite | 1 | 3.0 |
| print | 1 | 3.0 |
| export | 1 | 3.0 |
| book | 1 | 3.0 |
| download | 1 | 3.0 |
| pdf | 1 | 3.0 |
| printabl | 1 | 3.0 |
| version | 1 | 3.0 |
| add | 1 | 3.0 |
| modifi | 1 | 1.0 |
| april | 1 | 1.0 |
| creativ | 1 | 3.0 |
| common | 1 | 3.0 |
| attribut | 1 | 3.0 |
| sharealik | 1 | 3.0 |
| licens | 1 | 3.0 |
| addit | 1 | 1.0 |

| | | |
|---|---|---|
| appli | 1 | 1.0 |
| site | 1 | 1.0 |
| agre | 1 | 1.0 |
| privaci | 2 | 5.0 |
| polici | 2 | 5.0 |
| regist | 1 | 1.0 |
| trademark | 1 | 1.0 |
| wikimedia | 1 | 5.0 |
| foundat | 1 | 5.0 |
| profit | 1 | 1.0 |
| organ | 1 | 1.0 |
| disclaim | 1 | 3.0 |
| develop | 1 | 3.0 |
| cooki | 1 | 3.0 |
| statement | 1 | 3.0 |
| mobil | 1 | 3.0 |
| power | 1 | 2.0 |
| mediawiki | 1 | 2.0 |
| http | 1 | 0.0 |
| org | 1 | 3.0 |
| wiki | 1 | 1.0 |
| eyword | 1 | 1.0 |

# APPENDIX C

# VALUABLE MALICIOUS WEB PAGE RELATED WORDS

Selected a hundred keywords which increase maliciousness probability of a web page. The list is extracted from active features of MaxEnt-L1 model in order to create an imagination about features.

TableC.1: Sample 100 Valuable Malicious Web Page Related Keywords of MaxEnt-L1

| Keyword | Vector of Lambda |
|---|---|
| nda | 0.467202 |
| shqip | 0.444289 |
| seniorpeoplemeet | 0.314475 |
| dropbox | 0.314293 |
| herbal | 0.314269 |
| match | 0.273051 |
| ourtim | 0.263421 |
| success | 0.249248 |
| dhl | 0.242328 |
| factori | 0.232868 |
| multiplay | 0.231788 |
| pick | 0.217054 |
| center | 0.212807 |
| thoroughbr | 0.206742 |

| | |
|---|---|
| usaa | 0.204236 |
| showbiz | 0.198186 |
| fish | 0.19406 |
| outlook | 0.193962 |
| confirm | 0.189155 |
| telstra | 0.188968 |
| unlimit | 0.187963 |
| disk | 0.184646 |
| cape | 0.176581 |
| pin | 0.157502 |
| betti | 0.154905 |
| jnew | 0.153667 |
| westpac | 0.150679 |
| paypal | 0.139968 |
| million | 0.137915 |
| optician | 0.137914 |
| shorten | 0.137714 |
| longview | 0.137261 |
| chase | 0.136079 |
| fargo | 0.135525 |
| awar | 0.131012 |
| mail | 0.124968 |
| cours | 0.119689 |
| cgthb | 0.117949 |
| netsuit | 0.117026 |
| 404 | 0.115795 |
| aol | 0.115452 |
| signin | 0.115445 |
| balloon | 0.110458 |
| approv | 0.109981 |
| consult | 0.109836 |

| | |
|---|---|
| america | 0.108304 |
| matter | 0.10748 |
| bass | 0.106247 |
| constitut | 0.105126 |
| rug | 0.102291 |
| problem | 0.099736 |
| ogin | 0.097027 |
| drive | 0.095202 |
| treasuri | 0.095112 |
| american | 0.093814 |
| agreement | 0.093553 |
| inquir | 0.093073 |
| directori | 0.091781 |
| und | 0.091395 |
| epic | 0.089186 |
| prepar | 0.087076 |
| tini | 0.086954 |
| appl | 0.085909 |
| cottag | 0.084686 |
| txt | 0.083899 |
| client | 0.083161 |
| creat | 0.079323 |
| doc | 0.079007 |
| landscap | 0.078773 |
| suspend | 0.078603 |
| unavail | 0.077268 |
| facebook | 0.076898 |
| author | 0.075864 |
| phish | 0.074746 |
| lake | 0.069947 |
| attach | 0.069352 |

| | |
|---|---|
| translat | 0.068208 |
| label | 0.067806 |
| impot | 0.066952 |
| compt | 0.066659 |
| region | 0.064799 |
| consign | 0.064549 |
| hmrc | 0.063999 |
| webhost | 0.063907 |
| jaar | 0.063476 |
| china | 0.063287 |
| halifax | 0.061152 |
| barclai | 0.060541 |
| verifi | 0.060007 |
| sign | 0.059508 |
| friend | 0.057904 |
| bbb | 0.057826 |
| address | 0.057547 |
| collabor | 0.056547 |
| sandwich | 0.052889 |
| pharma | 0.051839 |
| exist | 0.051093 |
| set | 0.05024 |
| arnold | 0.049253 |
| builder | 0.048542 |

# APPENDIX D

## VALUABLE SAFE WEB PAGE RELATED WORDS

Selected a hundred keywords which increase safeness probability of a web page. The list is extracted from active features of MaxEnt-L1 model in order to create an imagination about features.

TableD.1: Sample 100 Valuable Safe Web Page Related Keywords of MaxEnt-L1

| Keyword | Vector of Lambda |
|---|---|
| dla | 0.482233 |
| twitter | 0.481712 |
| nie | 0.353226 |
| hous | 0.345134 |
| time | 0.310093 |
| softwar | 0.273936 |
| school | 0.273544 |
| video | 0.273133 |
| statement | 0.262144 |
| scienc | 0.259536 |
| big | 0.259177 |
| tool | 0.234872 |
| global | 0.232433 |
| market | 0.231081 |
| india | 0.226939 |
| offic | 0.226713 |

| | |
|---|---|
| profil | 0.224872 |
| tin | 0.210366 |
| project | 0.205417 |
| commun | 0.203801 |
| start | 0.201672 |
| win | 0.191704 |
| price | 0.184373 |
| bui | 0.182249 |
| new | 0.180038 |
| watch | 0.179421 |
| closet | 0.17612 |
| hillari | 0.169257 |
| nam | 0.165551 |
| gizlilik | 0.165215 |
| asia | 0.164323 |
| torrent | 0.164191 |
| real | 0.163422 |
| anim | 0.16151 |
| option | 0.160667 |
| march | 0.159983 |
| health | 0.155696 |
| legend | 0.150995 |
| film | 0.145166 |
| right | 0.142782 |
| accessori | 0.142751 |
| score | 0.141957 |
| ship | 0.141477 |
| complet | 0.140669 |
| sur | 0.139617 |
| love | 0.135238 |
| todai | 0.134747 |

| | |
|---|---|
| post | 0.133729 |
| forum | 0.133011 |
| gener | 0.130176 |
| cooki | 0.12848 |
| pour | 0.125677 |
| iyi | 0.12413 |
| check | 0.123534 |
| credenti | 0.12151 |
| academi | 0.118762 |
| driver | 0.117345 |
| iphon | 0.11449 |
| daha | 0.114387 |
| engin | 0.114104 |
| univers | 0.113937 |
| bet | 0.112604 |
| condit | 0.109852 |
| read | 0.109816 |
| solut | 0.109302 |
| alan | 0.108335 |
| game | 0.107017 |
| earn | 0.106589 |
| type | 0.105788 |
| babi | 0.105567 |
| offer | 0.105496 |
| food | 0.104128 |
| whoi | 0.103771 |
| deal | 0.103604 |
| publish | 0.101921 |
| sex | 0.101476 |
| jest | 0.100646 |
| parti | 0.100414 |

| | |
|---|---|
| blog | 0.099674 |
| releas | 0.099425 |
| test | 0.099101 |
| shop | 0.099088 |
| girl | 0.097818 |
| usa | 0.097782 |
| user | 0.096133 |
| intern | 0.095941 |
| stori | 0.09538 |
| euro | 0.094574 |
| cho | 0.094162 |
| send | 0.092425 |
| bni | 0.091871 |
| feedburn | 0.091506 |
| music | 0.091424 |
| dvd | 0.091046 |
| aso | 0.090909 |
| citi | 0.090187 |
| job | 0.089282 |
| turkei | 0.088475 |
| wordpress | 0.08832 |
| featur | 0.088074 |