Hindawi
Mobile Information Systems
Volume 2021, Article ID 8072779, 13 pages
https://doi.org/10.1155/2021/8072779

Research Article

Edge-Based Detection and Classification of Malicious Contents in Tor Darknet Using Machine Learning

Runchuan Li,[1] Shuhong Chen ,[1] Jiawei Yang,[1] and Entao Luo[2]

[1]*School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou 510006, China*
[2]*School of Electronics and Information Engineering, Hunan University of Science and Engineering, Yongzhou, Hunan 425199, China*

Correspondence should be addressed to Shuhong Chen; shuhongchen@gzhu.edu.cn

Received 2 September 2021; Accepted 25 October 2021; Published 22 November 2021

Academic Editor: Ke Gu

With the increase of data in the network, the load of servers and communication links becomes heavier and heavier. Edge computing can alleviate this problem. Due to a sea of malicious contents in Darknet, it is of high research value to combine edge computing with content detection and analysis. Therefore, this paper illustrates an intelligent classification system based on machine learning and Scrapy that can detect and judge fleetly categories of services with malicious contents. Because of the nondisclosure and short survival time of Tor Darknet domain names, obtaining uniform resource locators (URLs) and resources of the network is challenging. In this paper, we focus on a network based on the Onion Router (tor) anonymous communication system. We designed a crawler program to obtain the contents of the Tor network and label them into six classes. We also construct a dataset which contains URLs, categories, and keywords. Edge computing is used to judge the category of websites. The accuracy of the classifier based on a machine learning algorithm is as high as 89%. The classifier will be used in an operational system which can help researchers quickly obtain malicious contents and categorize hidden services.

1. Introduction

The Darknet has a huge amount of data. Edge computing can process massive data on terminal devices and transfer the processed results to the server, which alleviates the computing pressure of servers and the load of communication links [1]. Tor (The Onion Router) Darknet [2], which is also known as onion network and dark web, is a network using anonymous communication technology [3]. It is hard to access hidden services and obtain resources from it without using specific software or a proxy agency. Their sites are not only not indexed by Google or other standard search engines but also invalidate quickly [4]. Due to the good concealment of Tor Darknet a lot of illegal contents exit from it, such as drugs, guns, and hacking technology. After the outbreak of COVID-19, many medical products and supplies also appeared in the Darknet market [5] that is not good for the stability of the society.

AlQahtani and El-Alfy [6] conducted extensive research on anonymous technology and the onion network. The strong concealment of Tor network was illustrated, but there were some defects in the design and implementation of Tor hidden service technology [7–9]. Furthermore, due to the special network structure and the characteristics of hiding identities on both sides of communication, the improved onion network technology was also applied to other applications such as the Internet of Vehicles (IoV) ad hoc network [10]. Because of a large number of high-quality resources and the difficulty of obtaining them from the dark web, the mining and analysis of Tor network resources has been a major research hotspot in the academic community.

There are many research methods for Tor network resources. Web crawler technology can improve the efficiency of obtaining network resources. Iliou et al. [11] and Monterrubio et al. [12] proposed a general crawling framework for automatically obtaining web resources in

the Tor network. Wang et al. [13] and He et al. [14] proposed a method to identify anonymous traffic in the Tor network. Biswas et al. [15] proposed a method for automatic recognition of services in the Tor network based on perceptual hashing, which identified services only through snapshots of services. In addition to the above methods, the analysis of web page text is also a commonly used research method [16]. It can adopt methods of data mining and machine learning to analyze the data in hacker forums and Darknet markets and quickly screen out relevant threat intelligence [17–19].

Before analyzing the content of the dark web, it is necessary to obtain a large number of URLs and classify the topic of the website, so as to conduct targeted research. Kan and Nguyen Thi obtained the theme of the website by analyzing URLs [20]. However, due to the particularity of the domain name in Tor Darknet, the same method cannot be used to directly determine the category of the website.

Al Nabki et al. [21] and Spitters et al. [22] comprehensively analyzed the hidden services in the Tor network based on the web page content and classified themes of websites. Biryukov et al. [23] obtained hidden service descriptors through port scanning and classified the content. They found that the content of Tor hidden services is diverse. Al Nabki et al. [24] divided the Tor network addresses into 26 categories manually and selected nine categories to be applied to training three different supervised classifiers. However, they did not integrate the crawling and analysis process to form a visual interactive system. Buldin and Ivanov [25] used the K-nearest neighbor algorithm to identify four categories of Darknet web pages. Graczyk and Kinningham [26] classified products in anonymous marketplaces based on the support vector machine model. However, the accuracy of their classifying models is about 79%.

This study focuses on six distinct categories of Tor Darknet web sites. They are "Counterfeit money," "Counterfeit credit-cards," "Cryptocurrency," "Hacking," and "Drugs," respectively. These five categories do great harm to the society and are rich in resources in Tor Darknet. The rest of the classes are assigned to a 6th category which we called "Others."

To obtain the content of the Tor website, a program based on the Scrapy framework was designed. Then, we created a dataset to train a classifier based on the K-nearest neighbor (KNN) technique using a web-text preprocessing algorithm to extract features from webpages that can highlight the main theme. The optimal parameter for the KNN model was chosen using cross validation and Grid Search.

Finally, we improved the accuracy of the classifier to 89%, which is 10% higher than that of the classifier based on the support vector machine (SVM) algorithm in [26]. Furthermore, we designed a system to automatically crawl and classify the content of websites that exist in Tor Darknet. It will help researchers more easily obtain lots of content and identify categories of websites in Tor Darknet.

2. Proposed Model for Tor Darknet Resource Detection in Edge Computing

2.1. System Overview. A system combining the functions of detecting and analyzing Tor Darknet resources is designed in our work. It enables researchers to easily acquire the contents of Tor Darknet websites and classify websites into unknown categories. As shown in Figure 1, the system is split into three modules.

The edge computing module is primarily responsible for detecting and preprocessing the Tor Darknet content. Traditional cloud computing systems are experiencing network latency and bandwidth congestion as a result of the enormous data created by Internet of Things (IOT) devices. Edge computing was born out of the need for huge IOT devices to network and the high demand for real-time performance of their applications. Major applications, services, and data storage are sunk to the network's edge, bringing processing closer to the source of data. In the cloud computing paradigm, this will tackle issues such as excessive application delay and severe network load produced by enormous data being uploaded to the cloud data center for analysis. Web pages of URLs that are not categorized will be detected by the crawler. After preprocessing, the contents will be uploaded to a classifier, which relieves data processing pressure at the training module and decreases network transmission burden. The efficiency of accessing hidden services is limited due to the complexity of accessing the Tor network and the peculiarity of the routing protocol. Furthermore, the services generally have a short life cycle and are prone to malfunction. Therefore, web page content should be detected at edge devices, and the original data should then be processed into distinctive words that best describe the website category. The classification result may be returned more quickly and correctly after submitting the processed corpus to the classifier.

The classifier needs to be trained in the training module. To begin with, feature words in the dataset must be vectorized and weighted. The data is then split into two sets, a training set and a testing set, and they are used to train the classifier.

The output module is used to display the results. When unknown URLs are inputted, the crawler starts working and the data it collects is sent to the classifier for analysis. The system will display the classifier's results and a performance report. Furthermore, users have the option of saving any object and generating word clouds.

2.2. Domain Names Collection. In Tor Darknet, a domain name's complete format is "[digest].onion," which is made up of two parts: the first [digest] is a random string of numbers mixed with English, and the second is a uniform suffix of Tor links, jsaljfslj4sfd5ad.onion, for example. It will not show any results when we search sites with the suffix ".onion." Therefore, in order to classify the contents of Tor Darknet, domain names need to be obtained in various ways. In this paper, Tor domain names are collected in two ways: one is to collect them by Darknet directory sites; the other is

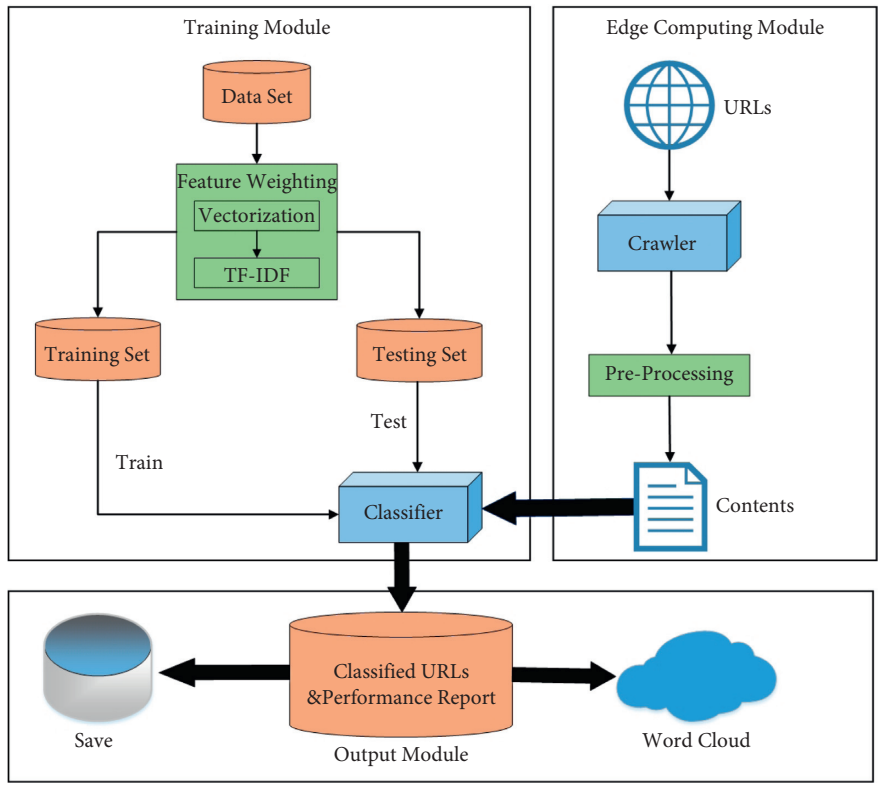

Figure 1: Tor Darknet detection and analysis system.

to use open datasets. URLs are mainly derived from a publicly available dataset called "Darknet Usage Text Addresses" (DUTA). It includes websites' domain names, categories, and language types. However, because of the lack of text content related to the site, we were supposed to design crawlers to obtain them. The content would be treated as much as possible to be the words which can reflect web categories. Simultaneously, we built a new dataset for training a classification model.

2.3. Communication Principle of Tor Darknet. The term "hidden service" (HS) is used to describe a website that runs on Tor Darknet. Aside from the uniqueness of its domain name, the way it communities with others is also interesting. You need to run Tor agent software locally to access the network. Figure 2 and Algorithm 1 depict the specific communication procedure between a client and a hidden service in the network.

The network's communication is established via a circuit made up of numerous routing nodes known as "Onion Routers" (OR). Following the launch of an HS, an Onion Agent (OA) will choose a router at random to serve as the Introduction Point Router (IPO), via which the hidden service will connect to the Tor network. The OA will then create a hidden service descriptor (HSD) including the IPO information, the timestamp, the HS public key, and other information, and upload it to a hidden service directory server (HSDS).

A user client connects to the network through the Onion Agent. The OA obtains the router information in the Tor network from a directory server (DS) and chooses the best-performing routers to construct a communication circuit. By default, the client connects to the HSDS via 3-hop OR. According to the domain name address sent by the client, the server queries the corresponding hidden service descriptor and returns, and then, the client resolves the IPO address. The client's OA chooses an OR as the Rendezvous Point Router (RPO) at random and transmits the RPO's information to the HS through IPO. RPO will serve as the hub for anonymized data exchange between the client and the HS. After learning the information of RPO, HS builds a 6-hop link to form a communication circuit and starts data transmission with the client.

Network configuration is necessary before the client can access Tor Darknet. There are two methods to connect to the network: one is to use Tor browser and the other is to configure the proxy environment.

The socks protocol is used for network communication. However, some crawling modules do not support this protocol, so they cannot directly obtain and parse the response returned by the site.

In this experiment, combining with the scrapy framework, the original configuration was modified and the proxy conversion software Polipo was used to convert the socks protocol into the HTTP protocol, so as to achieve the acquisition of Tor Darknet resources. Figure 3 shows the agency conversion:

2.4. Resource Detection for Tor Darknet. We created a crawling program based on Scrapy framework for Tor

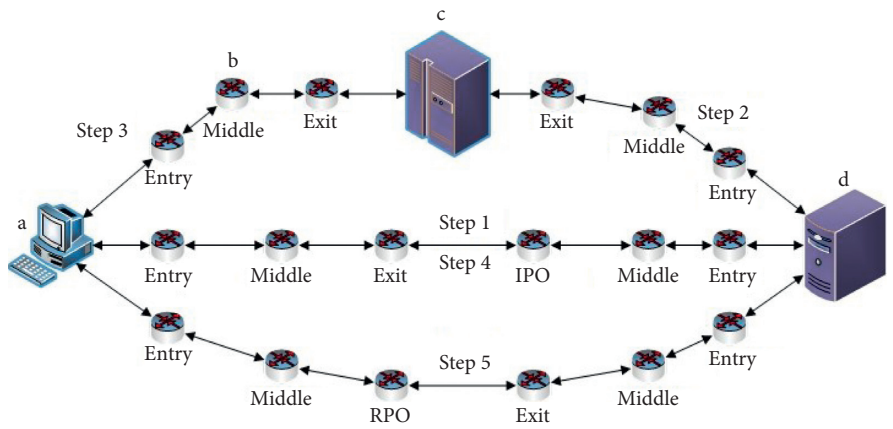

Figure 2: Communication between the client and HS. "a" is client, "b" is OR, "c" is HSDS, and "d" is HS. "Step n" is the order of communication.

Input: URL
Output: STATUS
(1) Client sends request to DS
(2) Set $R(OR[1], \ldots, OR[N]) \longleftarrow$ DS // DS returns the routing information to Client
(3) **for** $i = 1$ to $i = n$ **do**
(4) router whose performance is good will be selected
(5) **end for**
(6) STATUS = Client connects to HSDS and sends the URL to it
(7) **if** STATUS is failure **then**
(8) **return false**
(9) **else**
(10) Client get the information of IPO from HSDS
(11) **end if**
(12) **for** $i = 1$ to $i = n$ **do**
(13) Client selects a OR as RPO and sends its information to HS via IPO
(14) **end for**
(15) **return true** //data transmission can be started

Algorithm 1: Client connects to HS.

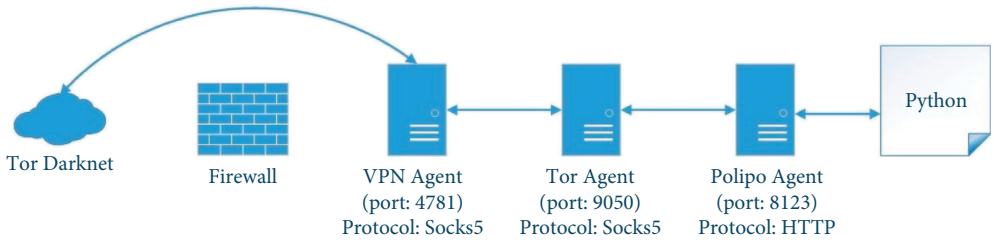

Figure 3: Agency conversion.

Darknet to accomplish automated access to resources due to the requirement to collect a huge number of URLs and the contents of each website.

The running process of the program is shown in Figure 4 and Algorithm 2.

A spider program reads all URLs in the list of "start_urls" and sends them to the engine, the center of the whole framework, which handles the data flow between components and triggers some operations. The engine will receive URLs to schedule, which will add URLs to the scheduling queue and wait for processing. After a URL is processed, the engine will receive a request sent by the schedule and trigger the downloader to work. After receiving a notification, the downloader will process a request according to the setting in downloader middleware and access Tor Darknet websites. The downloader will transmit a

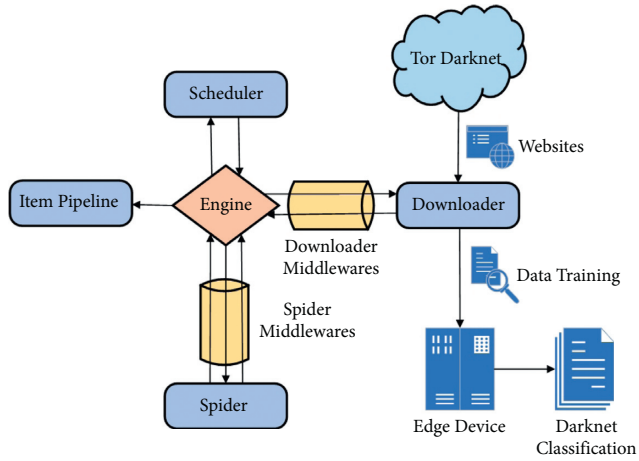

Figure 4: Framework of the crawling model.

processed request, namely, a response, to the spider for further processing. If the download fails, the engine will notify the schedule, then mark it, and reexecute the operation later.

When the spider receives a response, the "parse" function in the spider program will process the downloaded content according to the custom program. Then, items parsed in "parse" will be passed to the engine for further scheduling. The engine sends items to the item pipeline for data processing and storage in a file until the crawling of a URL is completed.

The URLs in the scheduling queue will then continue until all URLs have been processed. Scrapy does not repeat access to the site that has been visited, which solves the problem of multiple access to the same URL in the domain name set.

After the data is downloaded from Tor Darknet, the system will use the dataset to train a classification model and it will be downloaded to edge devices. Edge devices use the model to classify the content of unknown websites and finally obtain the categories of each website.

2.5. Data Preprocessing. The presentation of website page content and layout is realized through HTML code. After crawling a website to obtain texts, HTML elements in the texts need to be removed to filter out words. The data cleaning process is shown in Figure 5.

In Step 1, after the source code of a web page was downloaded, we parsed the HTML page content using "lxml.html.document_from_string" function from the LXML library. Then, "lxml.html.clean.cleaner().clean_html()" function was used to filter the script and HTML tags to obtain the text content displayed on the web page. There are many escape characters (ESC), carriage returns, tabs, and other meaningless characters in the content. Therefore, we combined them with Python's operation on string types to replace them with a space.

In Step 4, the corpus had no html tags and looked more like regular text. Word formats, on the other hand, were inconsistent. This would reduce the effect of selecting feature words and increase the dimension of feature space. We used the "casefold()" function to change all words to lowercase so that we could deal with them uniformly afterwards. All punctuation marks and Arabic numbers were filtered using an algorithm. We reduced inflection in words to their base forms using the Stemmer package, which helps to preprocess text, words, and documents for text normalization. We combine terms like "created" and "creates" into "create," for example.

When all of the corpora's formats were harmonized, they were expected to be processed further in Step 7. All the words that appear in the English stop word corpus were removed. These words, such as this, they, are, and so on, cannot represent any characteristics of websites. The last step was to remove strings that were longer than twelve or shorter than two. These characters are odd terms that do not correspond to the website's subject.

The specific implementation method is shown in Algorithm 3. After the text content of each web page is cleaned, the corpus is integrated into a dataset containing URLs, categories, and key words. A machine learning algorithm, KNN, is then applied to such samples for the purpose of training a classifier in subsequent experiments.

3. Classification Model

For each type of website, there must be some words that highlight its characteristics. Therefore, after vectorizing the corpus, we combined the machine learning algorithm to train a classifier suitable for Tor Darknet websites.

3.1. Text Vectorization. Before the classification of web content, it is necessary to transform words based on text representation into a form that can be recognized and calculated. In other words, words need to be transformed into vectors and the calculation of similarity between text semantics is transformed into the calculation of distance between vectors.

If the correlation between words and web topic is measured directly according to word frequency, the

Input: Set *start_urls* $(u[1], \ldots, u[n])$
Output: Set *Content*
(1) Queue Q = Enqueue (*start_urls*)
(2) **while** Q not empty **do**
(3) q = Dequeue(Q)
(4) **if** q has not been processed **then**
(5) *reponse* = download (q)
(6) **else**
(7) CONTINUE
(8) **end if**
(9) **if** status of website = 200 **then**
(10) *items* ⟵ parse (*reponse*)
(11) **else**
(12) mark it and re-executed the operation later
(13) CONTINUE
(14) **end if**
(15) *Content* ⟵ *Content* ∪ {*items*}
(16) **end while**
(17) **return** *Content*

Algorithm 2: Data crawling.

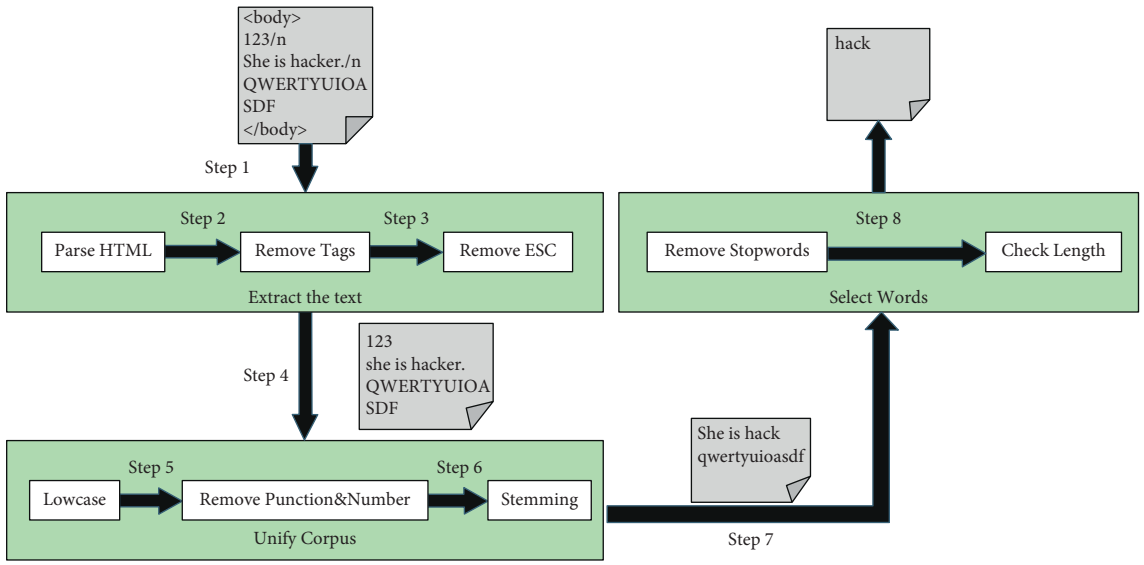

Figure 5: Data cleaning process.

Input: Web Set $D(D[0], \ldots, D[n])$
Output: Corpus set W $(W[0], \ldots, W[n])$
(1) **for** i = 0 TO $i = n$ **do**
(2) content = obtain the HTML content of $D[i]$
(3) use"lxml()"funtion to parse the content, then remove HTML tags, script andso on
(4) text = preserve the text content displayed on the page
(5) **for** *ch* in text **do**
(6) **if** *ch* = '\n' **or** *ch* = '\t' **then**
(7) *ch* = ''
(8) **end if**
(9) **end for**
(10) Lowercase all English words
(11) **for** *temp* in text **do**

Algorithm 3: Continued.

(12) **if** *temp* is a punctuation **or** a number **then**
(13) *temp* = ''
(14) **end if**
(15) **end for**
(16) PorterStemmer(text)//Unify all word formats
(17) word_list($wl[0], \ldots, wl[len1]$) = text.split('')
(18) **for** $i = 0$ TO $i = len1$ **do**
(19) **if** word_list[i] ∈ stopwords($sw[0], \ldots, sw[m]$) **or** 2 < len(word_list[i]) <12 **then**
(20) delete_wordlist[i]
(21) **end if**
(22) **end for**
(23) SET $W \leftarrow$ word_list($fw[0], \ldots, fw[len2]$).join('')//Words are concatenated to strings
(24) **end for**
(25) **return** W

Algorithm 3: Data cleaning algorithm.

measurement results will be related to the size of the web page and sometimes the words with a high degree of correlation cannot be really calculated.

For example, "the" appears frequently on one page, but it also appears on other pages, so it is not of much importance. "Hack" appears relatively infrequently on the page, but it exists only on that page. Therefore, it is more important than "the" in the page.

Therefore, a better weighted method should be adopted to calculate the words that are more relevant to the topic of the web page. Combined with the idea of TF-IDF (term frequency-inverse document frequency), this paper adopts a new weighting method to calculate the value of word vectors.

After accessing all the domain names, we get contents of each website and then build a web page set $W = (d_1, d_2, \ldots, d_k)$, where d_k represents the content contained in the kth web page.

Firstly, the frequency weighted method is used to count the occurrence times of all feature words in the corresponding web page text, as shown in formula (1), where $n_{i,j}$ represents the occurrence times of feature words t_i in web page d_j. And, t_i represents the ith word in the web page.

$$F_{tor}(t_i, d_j) = n_{i,j}. \tag{1}$$

The number of web pages containing the feature word t is represented by $G'_{tor}(t_i, d_j)$.

According to formula (2), where k is the size of the web page set and ε denotes the smoothing factor which ensures that the denominator is not zero, we calculate the inverse document frequency (IDF) of the feature word t in the web page d which is represented by $G_{tor}(t_i, d_j)$.

$$G_{tor}(t_i, d_j) = 1 + \log \frac{\varepsilon + k}{\varepsilon + G'_{tor}(t_i, d_j)}. \tag{2}$$

Then, we multiply the two values of $F_{tor}(t_i, d_j)$ and $G_{tor}(t_i, d_j)$ to obtain the TF-IDF value of the feature word t in the web page d, as shown in

$$H_{tor}(t_i, d_j) = F_{tor}(t_i, d_j) \times G_{tor}(t_i, d_j). \tag{3}$$

Finally, the TF-IDF values of all feature words are normalized according to formula (4). The denominator is the square root of the sum of the weighted values of all words in web page d, and the numerator is the weighted values of all feature words.

$$H_{\text{norm}}(t, d) = \frac{H_{tor}(t, d)}{\sqrt{\sum_{i=1}^{n} H_{tor}^2(t_i, d)}}. \tag{4}$$

After calculating the weighted values of all feature words, generally, the higher the value, the better the feature of the web page.

As shown in Figure 6, a web page set has two samples. Each value in rectangles is the weighted value of a word. The content of page A is "the hack hack" and page B is "the drug drug." According to the above formulas, the weighted value of each word in page A is calculated. Conspicuously, the weighted value of "hack" is higher than other words in the same page and "hack" is the word that can present the topic of the web page better.

3.2. Darknet Classification Model Based on KNN Algorithm. KNN (K-nearest neighbor) is a well-known supervised machine learning classification method with a well-developed theory and intuitive reasoning. After calculating the distance between "data to be categorized" and "samples of known category," the samples are classified by comparing the distance. Customize the K value, and choose K examples that are the most similar to the samples in the training set to be categorized. The proportion of the K sample categories is used to determine the target category of the sample to be categorized.

Assuming that the number of all feature words in web page set W is n, the text content of web page d_k is expressed as an n-dimensional vector $(w_1, w_2, \ldots, w_n)$, $d_k \in W$, where w_n represents the weight of the feature in the text. Calculate the distance between page x and page y according to

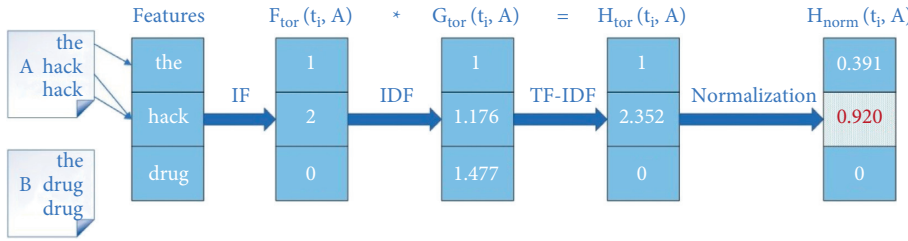

Figure 6: Text weighting process.

formula (5), where $w_i^{(x)}$ represents the weight of the ith feature in the text content of page x.

$$\text{Distance}(x, y) = \sqrt{\sum_{i=1}^{n} \left(w_i^{(x)} - w_i^{(y)}\right)^2}. \quad (5)$$

As shown in Figure 7, taking the two-dimensional space as an example, the square is a web page to be predicted. The triangle and circular are two different categories of websites. Calculate the distance of each sample point according to formula (5). Then, we take $K = 3$ samples closest to the square. Because the number of triangles in the circular area is the largest, the predicted square category is consistent with the triangle.

The implementation method of applying KNN algorithm to Tor Darknet classification is shown in Algorithm 4.

3.3. Model Optimization. In the KNN classification model, the hyperparameter K of the algorithm will affect the prediction result. When we choose a value of K too small, it is easy to lead to overfitting of the classification model. On the contrary, too large may lead to underfitting. However, there is no proper theory to guide the selection of the K value, which can be selected by manual experience and then determined by the cross-validation method.

Grid Search is a method of adjusting parameters. It traverses all candidate parameters and tries every possibility. The best-performing parameter is the best parameter we want to apply to the classifier. As shown in formula (6), m_i represents the model whose value of K is i, $acc(m_i)$ represents the accuracy of model i, and M_{best} represents the highest accuracy of the best model.

$$M_{best} = Max(acc(m_1), acc(m_2), \ldots, acc(m_i)). \quad (6)$$

Cross validation, also known as Cyclic Validation, is a method of model effect evaluation to avoid one-sided results caused by a single test. The concept of validation set is introduced to evaluate the accuracy of the model after tuning parameters, but it does not participate in training, which will more objectively evaluate the matching degree between the data outside the training set and the target attribute.

As shown in Figure 8, the dataset was divided into ten groups for ten model evaluations. Each time, one group of the dataset was taken as a validation set and the remaining nine groups were taken as the training set. As each validation set is different, the experiment will produce ten models and their accuracy is E_i. The average accuracy of ten models $acc(K)$ is the final accuracy of the classifier whose hyperparameter is K, as shown in formula (7).

Combined with the Grid Search and the 10-fold cross validation, the accuracy $acc(K)$ is applied to the calculation of formula (6) to select the model with the highest accuracy.

$$acc(K) = \frac{1}{10} \sum_{i=1}^{10} E_i. \quad (7)$$

4. Experimental Analysis

Python is the best choice for data crawling and analysis. It is favored by many developers because of its simplicity and powerful features. Therefore, in order to achieve our design goal, our experiments were developed based on Python 3.7 under Windows 10 according to the actual situation.

4.1. KNN Model Based on Frequency Weighting. In this experiment, we utilized the Grid Search to find an appropriate K on the scale of one to seven. Then, the results of KNN classification models with different parameters were evaluated in combination with the cross validation of ten folds. As shown in Figure 9, for models with the same K value, different partitions of the dataset would result differently, so the mean value was finally needed for comparison.

As shown in Figure 10, the black bar is the result of frequency weighting. The white bar is the result of TF-IDF weighting. Each bar represents the accuracy of the model with different values of K. The average accuracy is the highest when the value of K is three. Table 1 shows the evaluation report of the classification model whose K is three. The classification accuracy is only 0.78. This accuracy is too low to be applied to the system, and there will be a large error, so the model still needs to be improved.

4.2. KNN Model Based on TF-IDF Weighting. Figure 11 is the accuracy of 10-fold cross validation after weighting the feature vectors with TF-IDF.

As shown in Figure 10, after being weighted by TF-IDF, the results of classification models with different K values are compared by Grid Search. When the K value is four, the average accuracy of the model is the highest. The accuracy of this classifier reached 0.89. Table 2 shows the report of this model whose hyperparameter is four.

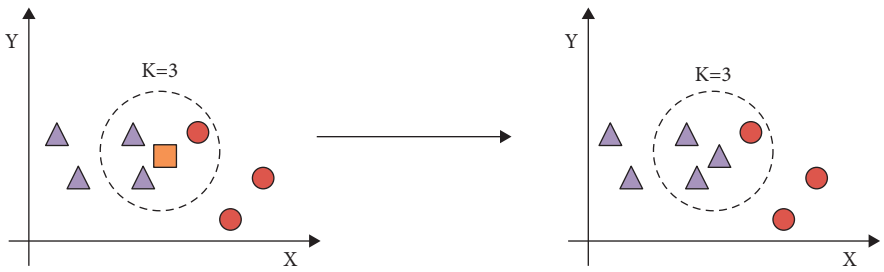

Figure 7: Classification principle of KNN algorithm.

Input: *TRAINSET* $(T[1], \ldots, T[n])$, Web page D to be predicted
Output: the category of D
(1) **for** $i = 1$ to $i = n$ **do**
(2) Calculate the Euclidean distance between each sample in *TRAINSET* and D
(3) *distance* = $(d[1], \ldots, d[n])$
(4) **end for**
(5) *sorted_dis* = (*distance*) //Sort in ascending order
(6) sample $(1, \ldots, k)$ = minum (*sorted_dis*) //select k samples that have the shortest distance
(7) set *classes* = count (*sample*) //Count the number of each category in k samples
(8) *category* = max (*classes*) //The category with the largest output number is the predicted category
(9) **return** *category*

Algorithm 4: KNN classification algorithm.

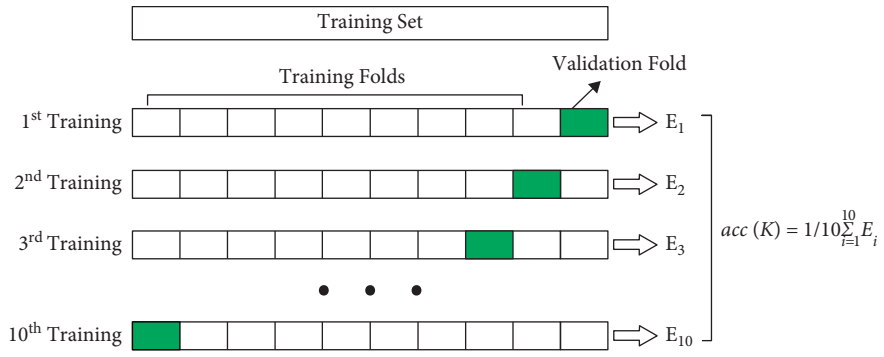

Figure 8: Ten-fold cross validation.

As shown in Table 3 and repeated experiments, the classification model after TF-IDF feature vectorization has higher accuracy and the performance is optimal when the hyperparameter K is four. Therefore, this model will be used in the subsequent system for its analysis function.

4.3. Comparison of Relevant Research Methods and Results. As shown in Table 4, the study in [24] manually marked 26 categories of domain names and selected 9 categories to apply in the training of the classification model. It adopted a Naive Bayes (NB) model based on TF-IDF weighting. Finally, the accuracy of cross validation of the model is 0.86.

The study in [25] categorized four types of illegal web pages based on the KNN algorithm. The accuracy for the test sample is 0.80. The work in [26] designed a model to automatically categorized products for anonymous marketplaces. It extracted features using TF-IDF and then selected features from the text using principal component analysis. It categorized 12 product categories using the SVM method, with a model accuracy of up to 0.79.

In this paper, we categorized six types of websites and four was finally determined as the optimal hyperparameter of the classification algorithm by combining Grid Search with cross validation. The data was applied to train the KNN classification model based on this parameter, and we improved the accuracy to 0.89 finally.

4.4. System Function Test. A file is selected to store the domain names after the network environment has been set

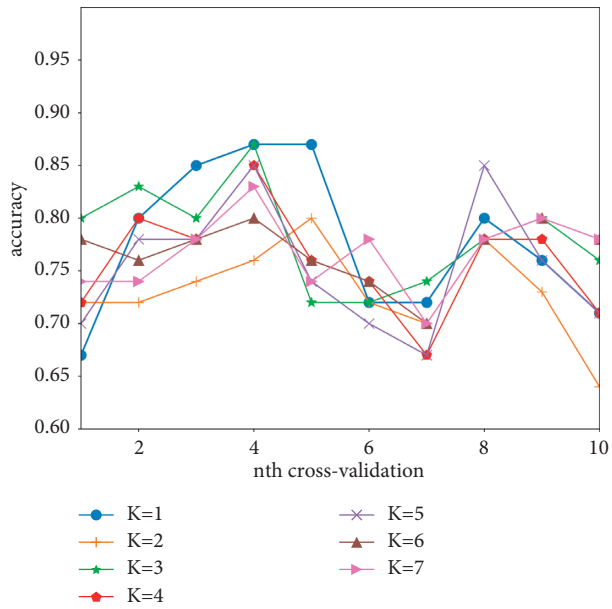

Figure 9: Ten-fold cross validation with frequency weighting.

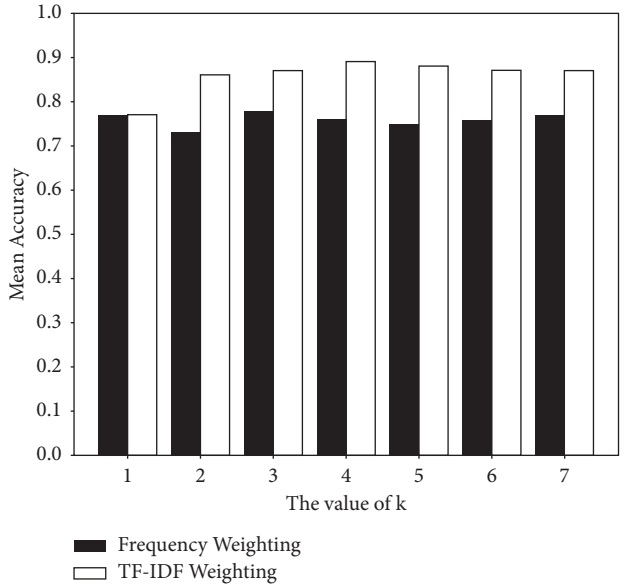

Figure 10: Cross-validation accuracy of model based on different weighting methods.

Table 1: Report of the KNN model which is weighted by frequency.

	Precision	Recall	F1-score	Support
Counterfeit credit-cards	0.76	0.87	0.81	30
Counterfeit money	1.00	0.83	0.91	12
Cryptocurrency	0.93	0.79	0.85	47
Drugs	0.78	0.67	0.72	21
Hacking	0.30	0.70	0.42	10
Other	0.84	0.78	0.81	63
Accuracy			0.78	183
Macro	0.77	0.77	0.75	183
Weighted avg	0.80	0.78	0.80	183

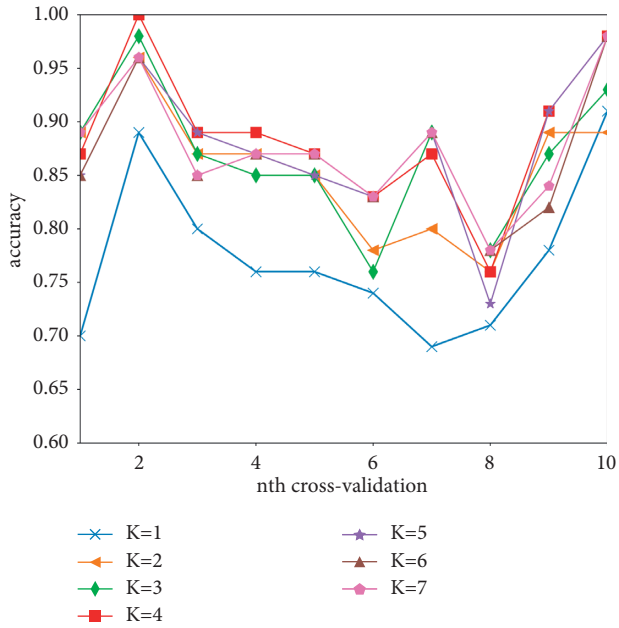

Figure 11: 10-fold cross validation with TF-IDF weighting.

Table 2: Report of the KNN model which is weighted by TF-IDF.

	Precision	Recall	F1-score	Support
Counterfeit credit-cards	0.88	0.90	0.89	31
Counterfeit money	0.33	0.50	0.40	4
Cryptocurrency	1.00	0.89	0.94	38
Drugs	0.89	0.73	0.80	22
Hacking	1.00	0.77	0.87	13
Other	0.87	0.96	0.91	76
Accuracy			0.89	184
Macro	0.83	0.79	0.80	184
Weighted avg	0.90	0.89	0.89	184

Table 3: Comparison of the best performance of KNN models based on different methods and sets.

	Frequency	TF-IDF
Test set	0.80	0.89
Validation set	0.78	0.89

Table 4: Comparison of methods and results between this paper and others.

Author	Accuracy	Method	Category
Al Nabki et al. [24]	0.86	NB	9
Buldin and Ivanov [25]	0.80	KNN	4
Graczyk and Kinningham [26]	0.79	SVM	12
Li et al.	0.89	KNN	6

up. After that, it will be able to access all domain names. The deactivated and visited domain names will also not be used again.

Crawling contents will be showed in the display area of classification results, as shown in Table 5. The contents of data are showed in part.

Table 5: Partial result after the system completed the detection in the Darknet.

Category	Url	Data
Cryptocurrency	http://grams7yngnpr5rzf.onion	helix light gram learn lingo buy
Counterfeit credit-cards	http://aaaajqiyzj34rhjm.onion	bring dream life low price
Counterfeit money	http://countervcgcdjp2y.onion/	counterfeit usd qualiti usd
Counterfeit money	http://fixedlwgc3burzts.onion	match european nation minor
Cryptocurrency	http://22222222gib3ywf6.onion	bitcoin quotation video onion site
Cryptocurrency	http://5ifblitg2ywjjo2fgt.onion	sourc bitcoin mixer quotation
Cryptocurrency	http://3fjgpldvlrd7k7im.onion	onion dir adult oniondir
Cryptocurrency	http://btce7mfnhnpdkg5k.onion	cost video xonion onion porn site
Drugs	http://napuzdankhadou3e.onion	dank hank weed logo gram amnesia haze sample
Drugs	http://smokerhxeb3tc6cy.onion/	smoke finest organ cannabi buy weed
Hacking	http://hackero5xh5qtrrzvmma.onion/	rentahack hire hacker job imagin ddo
Hacking	http://beast7ruvpc3qjhv.onion	itbazt hackasaserv hack ransomwar malwarew
Hacking	http://tinhat233xymse34.onion	darknet message hack broken statist unpack introduct setup
Hacking	http://agenttoe2dlvxdei.onion	price hack servic e-mail account devic
Other	http://underdj5ziov3ic7.onion	tor commun perman move onion onion deprec
Other	http://a64r6szrpegnggoj.onion	cryptostorm commun forum cryptostorm forum
Other	http://darkw4u3xkeb5pzn.onion	eva franco matt dark content dark content
Other	http://iz56hciijqh5uh5u.onion	celebr underground celebr underground bitcoi
Other	http://xujnrmw3lkpyj57r.onion	share news casino game multiplay game poker

Figure 12: Word cloud.

As shown in Figure 12, any domain name can be selected on the system interface and the system will automatically generate and display the word cloud images of such websites.

This function is helpful for researchers to intuitively obtain website feature words and create better visual effects.

5. Conclusions

Hidden services generally only retain service status for a limited amount of time in order to avoid being tracked, resulting in frequent domain address changes and a short lifespan. Because it requires special software and data must be encrypted and decrypted as it goes via each node on the communication circuit, access to Tor Darknet is sluggish.

For the reasons stated above, manually classifying websites creates a significant amount of labor. Furthermore, the pertinence is so low that it is impossible to rapidly reach a certain domain name type.

As a result, we created a visual interactive system based on edge computing that may assist researchers in swiftly obtaining content and classifying website categories. The classifying model's accuracy is as high as 89%, which will increase researchers' efficiency in identifying illegal websites and is of significant practical use. Furthermore, there are only six categories in this experiment due to the limited number of domain names already gathered. In the future, we will collect more domain names of other categories and obtain the feature words of different categories to expand the dataset which will contain more website categories.

Data Availability

The dataset used in this experiment is DUTA-10k, which is a classic open-source dataset for collecting Darknet addresses and can be downloaded from related websites. The dataset used in this experiment is downloaded from the GVIS platform (http://gvis.unileon.es/dataset/duta-darknet-usage-text-addresses-10k/).

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China under Grant 2020YFB1005804, in part by the National Natural Science Foundation of China under Grants 61632009 and 62172159, in part by the Guangdong Provincial Natural Science Foundation under Grant 2017A030308006, and in part by the Natural Science Foundation of Hunan Province under Grant 2021JJ30294.

References

[1] J. Bellendorf and Z. Á. Mann, "Classification of optimization problems in fog computing," *Future Generation Computer Systems*, vol. 107, pp. 158–176, 2020.

[2] S. Paul, D. Roger, and N. Mathewson, "Tor: the second-generation onion router," in *Proceedings of the Usenix Security Symposium*, pp. 303–320, San Diego, CA, USA, August 2004.

[3] M. Edman and B. Yener, "On anonymity in an electronic society," *ACM Computing Surveys*, vol. 42, no. 1, pp. 1–35, 2009.

[4] G. Owenson, S. Cortes, and A. Lewman, "The darknet's smaller than we thought: the life cycle of tor hidden services," *Digital Investigation*, vol. 27, no. 17–22, 2018.

[5] R. Broadhurst, M. Ball, and C. J. Jiang, "Availability of COVID-19 related products on tor darknet markets," *Australasian Policing*, vol. 12, no. 3, pp. 8–13, 2020.

[6] A. A. AlQahtani and E.-S. M. El-Alfy, "Anonymous connections based on onion routing: a review and a visualization tool," *Procedia Computer Science*, vol. 52, pp. 121–128, 2015.

[7] Q. Tan, Y. Gao, J. Shi, X. Wang, B. Fang, and Z. Tian, "Toward a comprehensive insight into the eclipse attacks of tor hidden services," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1584–1593, 2018.

[8] A. Biryukov, I. Pustogarov, and R.-P. Weinmann, "Trawling for tor hidden services: detection, measurement, deanonymization," in *Proceedings of the 2013 IEEE Symposium on Security and Privacy*, pp. 80–94, IEEE, San Francisco, CA, USA, May 2013.

[9] T. Elahi, K. Bauer, M. AlSabah, D. Roger, and I. Goldberg, "Changing of the guards: a framework for understanding and improving entry guard selection in tor," in *Proceedings of the 2012 ACM Workshop on Privacy in the Electronic Society, WPES '12*, pp. 43–54, New York, NY, USA, October 2012.

[10] M. Sayad Haghighi and Z. Aziminejad, "Highly anonymous mobility-tolerant location-based onion routing for vanets," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 2582–2590, 2019.

[11] C. Iliou, K. George, T. Tsikrika, S. Vrochidis, and I. Kompatsiaris, "Hybrid focused crawling on the surface and the dark web," *EURASIP Journal on Information Security*, vol. 2017, no. 1, pp. 1–13, 2017.

[12] S. Monterrubio, J. Naranjo, L. Lopez, and A. Caraguay, "Black widow crawler for tor network to search for criminal patterns," in *Proceedings of the 2021 Second International Conference on Information Systems and Software Technologies (ICI2ST)*, vol. 1, pp. 108–113, Los Alamitos, CA, USA, March 2021.

[13] L. Wang, H. Mei, and V. S. Sheng, "Multilevel identification and classification analysis of tor on mobile and pc platforms," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 2, pp. 1079–1088, 2020.

[14] G.-F. He, M. Yang, J.-Z. Luo, and L. Zhang, "Online identification of tor anonymous communication traffic," *Ruanjian Xuebao/Journal of Software*, vol. 24, no. 3, pp. 540–556, 2013.

[15] R. Biswas, V. González-Castro, E. Fidalgo, and E. Alegre, "Perceptual image hashing based on frequency dominant neighborhood structure applied to tor domains recognition," *Neurocomputing*, vol. 383, pp. 24–38, 2020.

[16] P. Kaur, "Web content classification: a survey," *International Journal of Computer Trends and Technology*, vol. 10, no. 2, pp. 97–101, 2014.

[17] S. Anna, A. Bessi, S. Damodaran, P. Shakarian, K. Lerman, and E. Ferrara, "Early warnings of cyber threats in online discussions," in *Proceedings of the 2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 667–674, IEEE, New Orleans, LA, USA, November 2017.

[18] D. Rhumorbarbe, L. Staehli, J. Broséus, Q. Rossy, and P. Esseiva, "Buying drugs on a darknet market: a better deal? studying the online illicit drug market through the analysis of digital, physical and chemical data," *Forensic Science International*, vol. 267, pp. 173–182, 2016.

[19] M. Ball and R. Broadhurst, "Data capture and analysis of darknet markets," 2021, https://ssrn.com/abstract=3344936.

[20] M.-Y. Kan and H. O. Nguyen Thi, "Fast webpage classification using URL features," in *Proceedings of the 14th ACM International Conference on Information and Knowledge Management, CIKM '05*, pp. 325-326, New York, NY, USA, October 2005.

[21] M. W. Al-Nabki, E. Fidalgo, E. Alegre, and L. Fernández-Robles, "ToRank: identifying the most influential suspicious domains in the Tor network," *Expert Systems with Applications*, vol. 123, pp. 212–226, 2019.

[22] M. Spitters, S. Verbruggen, and M. Van Staalduinen, "Towards a comprehensive insight into the thematic organization of the tor hidden services," in *Proceedings of the 2014 IEEE Joint Intelligence and Security Informatics Conference*, pp. 220–223, IEEE, Washington, DC, USA, September 2014.

[23] A. Biryukov, I. Pustogarov, F. Thill, and R.-P. Weinmann, "Content and popularity analysis of tor hidden services," in *Proceedings of the 2014 IEEE 34th International Conference on Distributed Computing Systems Workshops (ICDCSW)*, pp. 188–193, IEEE, Madrid, Spain, June 2014.

[24] M. W. Al Nabki, E. Fidalgo, E. Alegre, and I. de Paz, "Classifying illegal activities on tor network based on web textual contents," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pp. 35–43, Valencia, Spain, April 2017.

[25] I. D. Buldin and N. S. Ivanov, "Text classification of illegal activities on onion sites," in *Proceedings of the 2020 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, pp. 245–247, Moscow, Russia, January 2020.

[26] M. B. Graczyk and K. Kinningham, "Automatic product categorization for anonymous marketplaces," 2015, http://cs229.stanford.edu/proj2015/184_report.pdf.