Dark Web Text Classification by Learning through SVM Optimization

Ch A. S. Murty
Centre for Development of Advanced Computing (C-DAC), Hyderabad, India
Email: chasmurty@cdac.in

Parag H. Rughani
National Forensic Sciences University, India
Email: parag.rughani@nfsu.ac.in

***Abstract*—The Darkweb has become the largest repository of unauthorized information compared to the surface web because of its benefit of anonymity and privacy. With these anonymity and privacy features, the dark web is also becoming a safe place for illegal activities and hence an increase of dark web usage and size of the onion-based URLs. With the increasing use of dark web users, it is the need for cybercrime investigators across the globe to classify dark web data for understanding various illegal activities to control and categorize URLs hosting such illicit activities with feature engineering. In this research, the Support Vector Machines (SVM) algorithm is used to understand the algorithm's efficiency for a proposed model to classify dark web data with optimization techniques. Text-based keywords from more than 1800 websites were collected by applying feature engineering techniques and the system's performance was evaluated with the SVM approach. The results are very encouraging as the Precision, Recall, and F-measure values are 0.83, 0.90 & 0.96 achieved with a dataset of 1800 URLs.**

***Index Terms*—Darkweb, SVM, classification, Darkweb content classification**

I. Introduction

The Internet consists of tens of thousands of interconnected networks globally run by service providers, individual companies, universities and governments. The Internet ecosystem is supported by many organizations and communities that help to grow by sharing information[1]. The hosted information is accessible through browsers like Google Chrome, Firefox, Safari and others. The surface web is defined as a portion of the world wide web that is readily available and searchable to the general public through browsers and search engines like Google, Yahoo, Bing and others. On the other hand, the data in the deep web is not accessible and is also not indexed. The content in the deep web is only accessed by the respective owner and authorized users of that content, e.g., e-mails in your mailbox.

The remaining hidden within the Deep Web there is yet another shadowy Internet space called the "Dark Web" which cannot be reached by search engines and can only be accessed or browsed using anonymized networks such as The Onion Router (Tor), The Invisible Internet Project (IP2), Freenet, and dedicated applications corresponding to those networks[2]. So, the dark web is considered as one segment along with the deep web and surface web over the Internet. The dark web services [3] were initiated and developed in 2004, and in the last few years, the number of hosting services on the dark web and respective user's access increased exponentially. With the exponential growth of the dark web, text categorization of the dark web pages has become an essential technique for handling and organizing text data for cybercrime investigation [1]. The aim of this research is to understand the behavior of data and transactions in the dark web.

The classification of web text data with the text keywords saves time in searching, aggregating information, and managing data and unsupervised methods [2]. The visualization of the content in the dark web is also one of the ways that helps in classifying content [3].

Text Classification is defined as the automatic process to assign text to one or more predefined categories [4], [5]. One of the approaches used in text classification is based on Deep Learning [6]. In general, when we talk about supervised learning techniques, the web pages and URLs are labeled into a predefined class based on their content with different thresholds [7]. Text classification is applied in various fields such as information behavior, retrieval, filtering features for dark web data and others [8]. Majority of these techniques often use full-text content of the dark web text data to perform classification [9], which means that the classification process must deal with a large number of features. Users often use a set of keywords to search the dark web as keywords can express the core content, the crux of the hosted dark web page and can segregate the web page [10]. With a set of keywords

Manuscript received October 30, 2021; revised May 12, 2022; accepted May 16, 2022.
[1]The Internet ecosystem at https://www.internetsociety.org/internet/ecosystem
[2]Nikkei TECH, "Dark Web", http://tech.nikkeibp.co.jp/atcl/nxt/column/18/00178/040600011/
[3]The TOR Network, https://www.torproject.org, Tor Metrics, "Servers - Tor Metrics", https://metrics.torproject.org/networksize.html

doi: 10.12720/jait.13.6.624-631

representing the specific topic one can use these keywords to classify the new web URLs hosted in the dark web [11].

For a given set of text keywords and predefined categories, an estimation model [12] can be defined as $Y=f(X,\theta)+\in$ for a set of documents. The X is a suitably chosen text keyword (e.g., cannabis from the specific document in Drugs category), θ is a set of parameters associated with the class for classifier 'f' that need to be estimated and ∈ is the error of the classification class with that model. The error added that f is just an estimation of the actual but unknown function h such that Y = h(X). Hence, the smaller ∈ is, the more influential the classifier 'f' is. Once the classification model is ready, it can predict the category of new text.

The process of text classification from the dark web consists of the following six interrelated steps.

1) Text pre-processing
2) Text Representation or Transformation
3) Dimensionality reduction
4) Selection and Application of Classification Techniques:
5) Classifier evaluation, and validation

AI based classification of textual data from web-space is highly dependent on the process by which training data set is generated. While, it could be easier to collect e-content in the form of hosted documents and webpages from the surface web, it is highly challenging in the dark web [13], [14], for example, predicting fraudster e-mail based on their transactions [15]. Some of the other challenges pertaining to classification of textual data on dark web are discussed below.

The traditional classification based on Uniform Resource Locator (URL) addresses itself without collecting any text data from web sites is not possible in the dark web [16]. The dark web-based URLs are with an onion extension derived from a public key and self-certifying as TOR daemon, which makes it difficult to estimate the type of a website by having a URL address.

The scraping of data in the dark web also needs skills and software. The nature of encrypted communication in the dark web forces users to register for every web page, which is a tedious job for scraping. The websites on the dark web sites are hosted temporarily and many times they are hosted only for a few transactions. Further, some website owners host their content at multiple addresses, and randomly host one of those addresses for specific transactions.

In order to address the above challenges, we focused on designing and building a process that can be easy to train and can be adapted to different models for building a classification framework for dark web data. For effective classification performance, the following research contributions were made.

1) A custom dataset was created as a part of this research. Two methods were used for scraping web pages in the dark web, the first one collected keywords from all pages from URL domains and the second one, dumped text data of entire URL's domain and collected keywords from that domain.
2) To achieve this, a dataset of keywords from 1800 Dark Web URLs with predefined categories after manual labeling was created.
3) A classification framework for testing the performance of various models in identifying and predicting a given URL for what type of content it is hosting is proposed as the final outcome of this work.

II. Related Work

The classification of data from onion domain URLs based on linguistic information has been an approach followed by many researchers. Also, some researchers worked on text data from the URL itself by dividing it into words, characters in the URL as tokens for classification. The adoption of characters in URL as tokens for text classification is considered a faster method for text classification as it does not require any content in the respective website [17] and also proved to perform text classification of URL's text with individual binary classifiers and then combined via boosting into meta-binary classifiers with the achievement of F-Measure between 80 to 85 and a precision of 86. Concerning the dark web classification. Daniel Moore and Thomas Rid proposed TOR hidden service classification for an Automatic product categorization for anonymous marketplaces [18]. Initially, they experimented with 5000 TOR onion-based web sites as a sample and segregated them into 12 categories using an SVM classifier [19]. Graczyk proposed pipelines for the classification of the black market named Agora on Darknet [4]. They are classified into12 categories using TF-IDF for feature selection and PCA for feature selection in their pipeline architecture of SVM classification with an accuracy of 0.79. Takatoma proposed a system based on unique keywords by extracting from any web site and segregating into a specific class with automation, and classified tourism websites into different categories [20].

The classification of collected keyword texts with dark web URLs data starts with segregation into a fixed number of predefined categories. The collected keywords from hosted URL pages can be in multiple, exactly one, or no keyword for a particular category. The objective is to classify content from dark web URL Pages and perform assignments automatically to a predefined category. This process is in the general part of a supervised learning problem. Since some keywords may overlap with categories so that each category is treated as a separate binary classification problem.

III. Methodology

A. Data Collection

The keyword extraction was implemented in two ways — The first part with an automated script, which directly collected only keywords from the dark web pages with the supported environment by having Tails OS, SQL DB browser, Selenium, Beautiful Soup and Python scripts, and an overlay of VPN (Virtual Private Network) solutions to access the dark web.

The scripts were mainly used for crawling / scraping and automating the collection of random keywords of a particular website's data for classification, clustering and analysis. For efficient data collection, we used custom scripts in addition to existing crawlers [21]. We also used HTtrack to dump the entire webpage as the Darknet has a surprisingly substantial visible part and is well-connected with hyperlinks like the traditional web. The dataset is prepared from 1805 URLs for 88,000 keywords by running the custom script written by us at a random interval on each day for 120 days. This procedure helped us in extracting various keywords like "drugs", "child porn", "weapons", "bitcoin", and others from respective dark webs.

The other method was used for dumping entire hosted data from the deep web website and collecting keywords from data in case the respective URL is not allowed to scrape the data due to security implementations [22]. Moreover, the availability of those onion URLs is another challenge and especially, in this case, the dumped data also became useful for the research. A basic step-by-step process is shown in the following Fig. 1.

Figure 1. Data collection from dark web URLs' pages.

B. Keyword/Feature Extraction

While collecting keywords information, the word stems work well that their ordering in keywords of the database is always of low importance for many tasks. However, this may lead to an attribute-based representation of collected text keywords. So, each distinct keyword ***(ki)*** corresponds to a feature ***(fi***) with the number of times the keyword ***(ki)*** occurs in a page or site as its attribute value. At the same time, the number of times occurrence of a keyword may have impacted a particular feature, but for avoiding large feature vectors, keywords considered into feature selection only if they occur at least five times and also if they are not part of stop-words suck "like", "and", "or" etc.

The text keywords were collected from around 1800 URLs and segregated into ten different categories considered as features (fi). The system was designed with automation to segregate in predefined classes and classify hosted data in a URL by using a machine learning algorithm for given data. The predefined ten categories are 'Hidden wiki', 'Drugs', 'Market', 'Pornography', 'Counterfeit', 'Currency', 'Data', 'Frauds', 'Forum', and 'Services', which are further categorized into subcategories. Further, the predefined category keywords were used to make 'sentences' for a specific repository. A repository of keyword sentences prepared for all ten different predefined categories was prepared for easy maintenance and classifying for large databases. The keyword sentences are those that have the highest rate given to the relevant sentence of keywords. However, there is always a possible chance of higher bias in such matches. To avoid such bias, we considered three models named TF-IDF, One Hot Code with multiple categories and Bag-of-Words (BOW), which helped us translate keywords from categorical to numerical for enhancing keywords dataset.

The One hot Code method which was utilized had ten different classes, e.g. ['Hidden wiki', 'Drugs', 'Market', 'Pornography', 'Counterfeit', 'Currency', 'Data', 'Frauds', 'Forum', 'Services']. Hence, to represent one-hot encoding for presence of 'Drugs', the 10-dimensional binary vector would be [0,1,0,0,0,0,0,0,0,0]. For multi-hot encoding the label-encoding to classes was used, thus having only a single number representing the presence of a class (e.g., 1 for 'drugs') and then converting the numerical labels to binary vectors. This representation is the middle way between label-encoding, where you introduce false class relationships (0 < 1 < 2 < ... <9, thus 'drugs' < 'market' < ... < 'Hidden Wiki') but only need a single value to represent the class presence and one hot-encoding. This method requires a vector of size n! to represent all classes but have no false relationships.

To increase the accuracy and avoid overfitting for a dataset, using all possible conventions for learning methods to classify the data. One of the popular statistical methods is the 'Information Gain' criterion and used in preparing a subset of features [23]. Also, the Inverse Document Frequency (IDF) method improves the performance for the feature selection by scaling the dimensions of vectors with the term frequency variant [24]. To abstract from different document lengths, each document features a vector normalized to unit length. The BOW (Bag of Words) method is a model for text representation that extracts the text and feature selection based on the word frequency. The TF-IDF is a statistical model that assigns weights for the keywords that emphasize the words that frequently occur on a given web page [25]. The BOW (Bag of Words) method on the other hand is a model for text representation that extracts the text and feature selection based on the word frequency. The TF-IDF is a statistical model that assigns weights for the keywords that emphasize the words that frequently occur on a given web page.

Based on the above, the important keywords related to a category are grouped into classes based on similarity of words or actions.

C. Selection of Classifier

As a part of the classification of the dataset of various categories, the SVM Classifier was selected for training the data and to understand the capability of the classifier to

adopt in building tools and technologies in classifying the deep web [26]. For all ten categories, a separate repository of keyword sentences prepared for the classification of the entire dataset into main classes for training the SVM Classifier [27].

The classification of text data by a support vector machine (SVM) algorithm accustomed to plot data as points in multi-dimensional space where each class defined for the dataset acts as one dimension of a particular coordinate for the dataset. Hyperplanes act as decision boundaries that classify data points into their respective classes in a multi-dimensional space. Data points falling on either side of the hyperplane is also attributed to different Classes. A hyperplane can act as a support vector to some extent in 1-dimensional space, a line in 2-dimensional space, and an idea in 3-dimensional space as shown in Fig. 2 for classification data. However, a line in 3-dimensional space is not a hyperplane and does not separate into two parts.

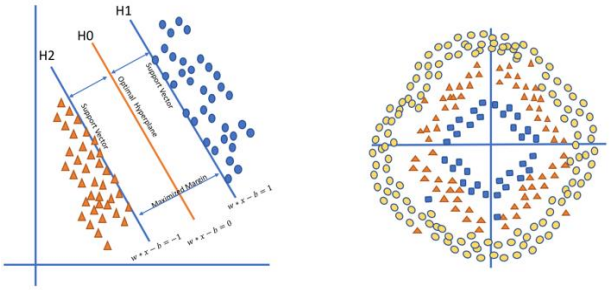

Figure 2. SVM linear and n-dimensional space.

SVM is assumed to be a large margin classifier where the distance between a line and vector space[4] of closest data points on either side is referred to as margin lines. The optimal line that has the higher-margin would act as the largest margin hyperplane. The margin calculated that as the perpendicular distance from the line to only the closest points. The points to the line are called support vectors. In this paper, we tried to maximize the margin of the classifier and try to understand the support vectors will impact the hyperplane position such that it helps us build an SVM model for dark web text data.

Support Vector Machines (SVMs) are powerful kernel methods for classification and regression tasks. If trained optimally, they produce excellent separating hyperplanes. The quality of the training for the SVM algorithm depends not only on the given training data but also on additional learning parameters, which are difficult to adjust in particular for unbalanced datasets.

D. Understanding Classifiers for SVM

The principle of Structural Risk Minimization [28] is a computational learning theory [29] and an inductive principle use of machine learning algorithms like learning Support Vector Machines algorithms. The thought of Structural Risk Minimization (SRM) is to look out a hypothesis, '***h***' that we are ready to guarantee that the bias is a minimum to rock bottom. The verity bias of '***h***' is the probability that '***h***' will make a bias on unseen and randomly selected test examples. There is a need for a position to connect the verity bias of a hypothesis '***h***', with the bias of '***h***' on the training set. and so, the complexity of high dimensional space. The Support Vector Machines with a hypothesis '***h***' expects and minimizes this bound on actuality bias by effectively and efficiently controlling the high dimensional space[5]. We define the hypothesis function '***h***' as:

$$h(x_i) = +1, if w * x + b \geq 0 \quad (1)$$

$$h(x_i) = -1, if w * x + b < 0 \quad (2)$$

The point above or on the hyperplane will be classified as class +1, and the point below the hyperplane will be classified as class -1. The distance from a point (x_0, y_0) to a line $A_x + B_y + c = 0$ is:

$$d = \frac{Ax_0 + By_0 + c}{\sqrt{A^2 + B^2}} \quad (3)$$

So, the space between H_0 and H_1 is then:

$$\frac{|\boldsymbol{w}*\boldsymbol{x}+\boldsymbol{b}|}{||\boldsymbol{w}||} = \frac{\mathbf{1}}{||\boldsymbol{w}||} \quad (4)$$

In order to maximize the margin, we thus must minimize ||w|| with the condition that there aren't any data points between H_0 and H_1.

The above SVM formulation is named Hard Margin SVM which suggests that an SVM is extremely rigid in classification and tries to figure extremely well within the training set, causing overfitting. The overfitting problem with Hard Margin SVM doesn't tolerate outliers so it doesn't work with non-linearly separable data due to outliers. For optimization the constraints can be considered as:

$$y_i * (x_i + b) \geq 1 \text{ when } y_i = +1 \quad (5)$$

$$x_i * w + b \leq -1 \text{ when } y_i = -1 \quad (6)$$

$$for\ every\ example\ combined\ into\ y_i(x_i * w) \geq 1$$

For solving the optimization problem, the related constraints should be satisfied with outliers. The optimization will not be solvable in the case of some outlier example which makes any constraint not satisfied. There is a need to cater to this limitation by employing a variant called soft margin SVM. As the hard margin SVM works only with linearly separable data, the information will contain some noise and may not be linearly separable in practical cases. The soft margin SVM classifier works to minimize the expression of the shape.

$$\frac{\mathbf{1}}{\boldsymbol{n}} \sum_{i=1}^{n} max\,(0, 1 - y_i(x_i - b)) + \lambda||w||^2 \quad (7)$$

Focusing on soft margin classifiers by choosing a small value which may be sufficient for λ (lambda) yields

[4] https://heartbeat.fritz.ai/demystifying-support-vector-machines-python-875f28a1b26c

[5] https://stats.stackexchange.com/questions/23391/how-does-a-support-vector-machine-svm-work

linearly separable for hard-margin SVM classifiers. At the same time, there is no need to minimize the target function by choosing negative values of λ (lambda) with adding the constraints $\lambda_i \geq 0$. Also, adding a regularization parameter C to work out how important λ, should be, which implies what quantity is required to avoid misclassification for each training example. The regularization in optimization problem defined as:

$$w, b, \lambda \left[\frac{1}{2} \left| |w| \right|^2 + C \left(\sum_{i=1}^{n} \lambda_i \right) \right]$$

$$where\ y_i(x_i + b) \geq 1 - \lambda_{i,}\ i = 1,2, \dots, n \qquad (8)$$

Adding slack variables like (λ_i) to the constraints for the optimization problem and choosing a small value for lambda may yield the hard margin classifier for any linear classifiable computer file. The regularization parameter C supports how important λ should be. The larger value 'C' diminishes the importance of 'λ', whereas the smaller value of 'C' emphasizes the importance of λ. The SVM handles errors by controlling the value of C. If we set C to 0, there will not be any constraint that results in a hyperplane not classifying anything with your data.

The noise data is due to non-linear data where the soft margin SVM can handle the same to classify the data. It is better to use kernel tricks if the dataset is characteristically non-separable linearly. However, when we transform the two-dimensional data to a better dimension, such as three-dimension or maybe even ten-dime or higher dimension, we may find a hyperplane to separate the information in the dataset. The polynomial kernel defined as:

$$k(x_i, x_j) = (x_i * x_j + c)^n, where\ n\ is\ degree \qquad (9)$$

The kernel '**k**' contains two parameters as one of them is '**c**', which is relentless as another one is the degree of freedom '**n**'. In this case, the degree value with 1 makes the linear kernel. A higher value of the degree will choose more complex and might lead to an overfitting problem. The RBF (Radial Basis Function) kernel defined as:

$$k(x_i, x_j) = e^{-\gamma \left| |x_i - x_j| \right|^2} \qquad (10)$$

The RBF kernel is also called the Gaussian kernel. It will lead to a more complex decision boundary of the space. The parameter γ in RBF kernel having significance as a smaller value of γ will make the model behave sort of like a linear SVM. A higher value of γ will make the model heavier.

In general, the SVMs are universal learners and learn linear threshold function in their basic form. The SVMs can be used to learn polynomial classifiers, Radial Basis Function (RBF) networks, and three-layer sigmoid neural nets by a simple 'plug-in' of an appropriate kernel function. Another remarkable property of SVMs is the ability to learn independently to the dimensionality of the feature space. The SVMs measure the complexity of separating data based on margin for a given hypothesis, but not with the number of features. It means that we can generalize even in the presence of many features like 'Hidden wiki', 'Drugs', 'Market', 'Pornography', 'Counterfeit', 'Currency', 'Data', 'Frauds', 'Forum', 'Services' etc., and data is separable with a wide margin using functions from the hypothesis space.

The choice of suitable learning parameter is a pivotal step in obtaining support vector machines for any dataset. In general, the settings of these parameters are based on the grid-based search method [30]. The grid search is nothing but, for every parameter, a finite number of possible values is prescribed, and then all making all possible combinations of values and finding a suitable one that yields the best result. Therefore, the complexity of grid search grows exponentially with many parameters. One of the main reasons for the fact that typically only two learning parameters have been used in SVM, namely error toleration with maximization, and the width of the standard kernel [31], e.g., Gaussian kernel, the degree of a polynomial kernel[6].

IV. Implementing SVM in Python With Sklearn

We have used ***sklearn***, which is one of the most robust and useful Machine Learning (ML) python libraries for supervised and unsupervised algorithms [32]. Further, we imported the SVM module and created a support vector classifier object by passing the argument kernel as linear / Polynomial / RBF as SVC () function and tried to fit our model on train set using fit() function to perform prediction on test set using predict() function. Then we evaluated the model which can predict for given input text data. i.e. text keywords / sentences as input to validate and predict the class which belongs to.

```
X = dataset.values[independent]
y = dataset. values[dependent]
from sklearn.svm import SVC
SVM_DW_Classifier = SVC
(kernel= 'linear'/'Polynomial')
SVM_DW_classifier.fit (X_train, y_train)
prediction = SVM_DW_classifier. predict(X_test)
```

V. Results and Discussion

The dark web URL's data with ten categories collected for SVM experimental purposes are shown in Table I. Also, we used the dataset for a set of 10 experiments by splitting the dataset into 70:30 and 80:20 ranges for training and testing datasets for cross-validation. Further, 80% of the training set was used and tested with the remaining 20% URLs in each category in an integrated manner to test their illegal content in the Python environment.

[6]R. Herbrich, Learning Kernel Classifiers: Theory and Algorithms, MIT Press, Cambridge, MA, USA (2001).

TABLE I. NUMBER OF URLS IN TRAINING AND TESTING

Category	Number of URLs	Training Set	Testing Set (10%)	Training Set	Testing Set (20%)
Hidden wiki	132	119	13	106	26
Drugs	319	287	32	255	64
Market	411	370	41	329	82
Pornography	279	251	28	223	56
Counterfeit	121	109	12	97	24
Currency	109	98	11	87	22
Data	92	83	9	74	18
Frauds	123	111	12	98	25
Forum	87	78	9	69	18
Services	132	119	13	106	26
Total	1805	1625	181	1443	362

To evaluate the performance of a model or classifier with SVM, The Precision, Recall and F-measure values were calculated and are shown in Table II and Fig. 3.

TABLE II. PERFORMANCE OF SVM KERNEL ALGORITHMS FOR SAMPLE SIZE DATA IN RATIOS OF 80:20 & 70:30

Kernel	TS	P	R	F1	A
Linear	80:20	0.68	0.75	0.71	0.72
	70:30	0.73	0.67	0.70	0.70
Polynomial	80:20	0.72	0.71	0.71	0.73
	70:30	0.72	0.71	0.72	0.71
Gaussian	80:20	0.83	0.90	0.96	0.87
	70:30	0.83	0.84	0.84	0.83
Sigmoid	80:20	0.46	0.50	0.50	0.50
	70:30	0.53	0.47	0.50	0.51
RBF	80:20	0.80	0.86	0.83	0.83
	70:30	0.83	0.84	0.84	0.83

P: precision, R: Recall, F1: F1-Score and A: Accuracy

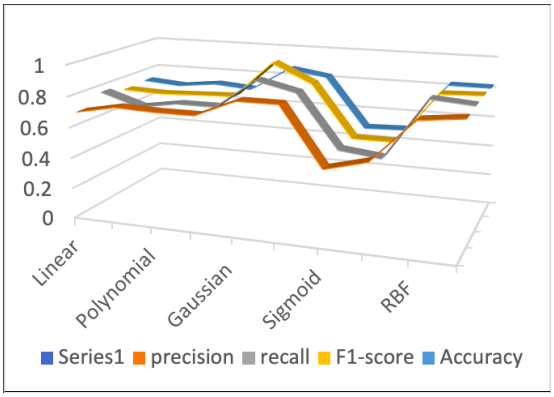

Figure 3. Performance of SVM algorithms.

The average of Precision Recall and F-measures of Gaussian kernel and RBF kernels **0.83, 0.90 & 0.96** and **0.83, 0.86 and 0.83** respectively were achieved. When we compared with sigmoid, linear and polynomial kernels for collecting the dark web dataset, the average accuracy of Gaussian Kernel and RBF Kernels are at 0.87 and 0.83 are achieved.

When we compared the performance of the different types of SVM kernels with dark web text data, It is observed that the Gaussian kernel achieved higher performance compared to the remaining kernel methods along with the linear kernel. Also, it is observed that the sigmoid kernel performs the worst as the reason may be that the sigmoid function returns only two values with '0' and '1'. Therefore, it is more suitable for binary classification problems. In this model, we considered multi classes and respective features with SVM algorithms.

We have also compared the Gaussian and RBF kernels, and it is observed that the Gaussian Kernel performed slightly better than the RBF cluster. The SVM classifier with the Gaussian kernel is simply a weighted linear combination of the kernel function computed between a data point and each of the support vectors where as in the RBF kernel defining the transformed features in terms of the original features of a data point leads to an infinite-dimensional representation and also the radial basis functions are set of functions which have same value at a fixed distance from a given central point.

Further Gaussian Kernel is also considered as a universal kernel which guarantees the global optimal predictor which helps in minimizing the estimation and approximation of errors. The aforementioned statement could be one of the reasons why the Gaussian kernel performed better. In our classifier the data set contains "n" distinct classes so the Gaussian kernels map as an infinite dimensional space. By making space with the "n" classes as large as possible, we obtain a Gaussian function of the distance between the classes in the dataset so that it can operate and keep growing indefinitely.

Considering the complex nature of dark-web for example where in a marketplace website you can find various subcategories like drugs counterfeits item etc. such an environment where there is very thin line between the text data when it comes to classification, such problem can be overcome by a features of Gaussian kernel i.e., supports the infinite complex models in to finite model and also, the selection of Gaussian smoothing and filtering helped in better results.

VI. CONCLUSION

This paper was introduced to determine the performance of various kernel functions of SVM Algorithm for the classification of text data hosted in the dark web which is further used for identifying illegal activities. This method does not rely on collecting a large number of illegal web pages to train the models, but only requires limited data in the form of keywords. The number of keywords collected are made into a predefined category and converted again in the form of statements and in turn to documents for classification. We applied the SVM model to effectively classify the hosted text data and to identify the illegal activities on the dark web and finally to check with the performances of models. For collecting as many suitable keywords, we used a custom script for scraping the data from URLs directly and also used tools like HTtrack to dump the entire dark website. The proposed system classifies text data purely based on keywords collected from the scraped dark web text data. We achieved an accuracy of the Gaussian Kernel with 0.87 and also performed a comparative analysis by comparing the accuracy with other kernel functions. Further also, compared the Precision, Recall and F-measure values. We have achieved the highest of Precision, Recall and F-

measure values i.e., 0.83, 0.90 and 0.96 respectively with this approach. The main success of SVM approach-based classification predicted well for any random keyword and classified into a predefined category. Also, it can support in monitoring the dark web behavior with text data of lesser URLs and may assist the investigators to track and block illegal sites. This classifier can be updated or extended to various machine learning models to improve the performance of models.

CONFLICT OF INTEREST

The authors have no conflict of interest to declare.

AUTHOR CONTRIBUTIONS

The authors confirm contribution to the paper as follows: Conceptualization and design by Ch. A. S. Murty and Parag. H. Rughani; data collection, research, analysis, interpretation of results and preparation of draft manuscript by Ch. A. S. Murty; review of results and approval of the final version of the manuscript by both the authors.

REFERENCES

[1] R. Rajalakshmi and C. Aravindan, "Web page classification using n-gram based URL features," in *Proc. 5th International Conference on Advanced Computing*, 2013, pp. 15-21.

[2] F. Liu, D. Pennell, F. Liu, *et al.*, "Unsupervised approaches for automatic keyword extraction using meeting transcripts," in *Proc. Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the ACL*, June 2009, pp. 620-628.

[3] S. Takaaki and I. Atsuo, "Dark web content analysis and visualization," in *Proc. Fifth ACM International Workshop on Security and Privacy Analytics*, 2019.

[4] K. Kinningham and M. Graczyk, "Automatic product categorization for anonymous marketplaces," *Comput. Sci.*, pp.1-6, 2015.

[5] V. Kobayashi, S. Mol, H. Berkers, *et al.*, "Text classification for organizational researchers: A tutorial," *Organizational Research Methods*, vol. 21, no. 3, 2018.

[6] J. Cai, J. Li, W. Li, *et al.*, "Deep learning model used in text classification," in *Proc. 15th International Computer Conference on Wavelet Active Media Technology and Information Processing*, 2018, pp. 123-126.

[7] Y. Matsuo and M. Ishizuka, "Keyword extraction from a single document using word co-occurrence statistical information," *Int. Journal on AI Tools*, vol. 13, no. 1, pp. 157-169, 2004.

[8] S. Lakhotia and X. Bresson, "An experimental comparison of text classification techniques," in *Proc. International Conference on Cyberworlds*, 2018, pp. 58-65.

[9] Q. D. Truong, H. X. Huynh, and C. N. Nguyen, "An abstract-based approach for text classification," in *Proc. International Conference on Nature of Computation and Communication*, 2016, pp. 237-245.

[10] E. H. Han, G. Karypis, and V. Kumar, "Text categorization using weight adjusted k-nearest neighbour classification," in *Proc. Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2001.

[11] T. C. T. Tran, H. X. Huynh, P. Q. Tran, *et al.*, "Text classification based on keywords with different thresholds," in *Proc. the 4th International Conference on Intelligent Information Technology*, February 2019, pp. 101-106.

[12] C. C. Aggarwal and C. Zhai, "A survey of text classification algorithms," in *Mining Text Data*, Springer, 2012, pp. 163-222.

[13] X. Qi and B. Davison, "Web page classification," *ACM Computing Surveys*, vol. 41, pp. 1-31, 2009.

[14] S. He, Y. He, and M. Li, "Classification of illegal activities on the dark web," in *Proc. the 2nd International Conference on Information Science and Systems*, March 2019, pp. 73-78.

[15] J. Shen, O. Brdiczka, and J. Liu, "Understanding e-mail writers: Personality prediction from e-mail messages," in *Proc. International Conference on User Modeling, Adaptation, and Personalization*, 2013.

[16] M. Kan, "Web page classification without the web page," in *Proce. the 13th international World Wide Web Conference on Alternate track papers & posters*, 2004.

[17] E. Byaken, M. Heninger, and I. Weber, "A comprehensive study of features and algorithms for URL-based topic classification," *ACM Transactions on Web*, vol. 5, no. 3, July 2011.

[18] D. Moore and T. Rid, "Crypto politik and the darknet," *Survival*, vol. 58, no. 1, pp. 7-38, January 2016.

[19] A. Sun and E. P. Lim, "Web classification using support vector machine," in *Proc. the 4th International Workshop on Web Information and Data Management*, 2002, pp. 96-99.

[20] T. Honda, M. Yamamoto, and A. Ohuchi, "Automatic classification of websites based on keyword extraction of nouns," in *Information and Communication Technologies in Tourism*, Vienna: Springer, 2006, pp. 263-272.

[21] S. Mirtaheri, M. Dinçtürk, S. Hooshmand, *et al.*, "A brief history of web crawlers," arXiv preprint arXiv:1405.0749, 20134

[22] M. Kan and H. Thi, "Fast webpage classification using URL features," in *Proc. the 14th ACM International Conference on Information and Knowledge Management*, 2005, pp. 325-326.

[23] T. Eitricha and B. Lang, "Efficient optimization of support vector machine learning parameters for unbalanced datasets," *Journal of Computational and Applied Mathematics*, vol. 196, no. 2, November 2006.

[24] G. Saltonand and C. Buckley, "Term weighting approaches in automatic text retrieval," *Information Processing and Management*, vol. 24, no. 5, pp. 513-523, 1988.

[25] N. Fei and Y. Zhang, "Movie genre classification using TF-IDF and SVM," in *Proc. the 7th International Conference on Information Technology: IoT and Smart City*, December 2019, pp 131-136.

[26] O. Akyıldız, "Information analysis and cyber crimes in deep web & dark web," in *Proc. 6th International Symposium on Digital Forensic and Security*, 2018, pp. 1-6.

[27] S. Tavara, "Parallel computing of support vector machines: A survey," *ACM Computing Surveys*, vol. 51, no. 6, February 2019.

[28] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, pp. 273-297, 1995.

[29] X. Zhang and F. Ren, "Improving SVM learning accuracy with Adaboost," in *Proc. Fourth International Conference on Natural Computation*, 2008, pp. 221-225.

[30] C. W. Hsu, C. C. Chang, C. J. Lin. (2003). A practical guide to support vector classification. [Online]. Available: http://www.csie.ntu.edu.tw/cjlin/papers/guide/guide.pdf

[31] G. A. Gray and T. G. Kolda. (2004). APPSPACK 4.0: Asynchronous parallel pattern search for derivative-free optimization. Sandia Report SAND2004-6391, Sandia National Laboratories, Livermore, CA. [Online]. Available: http://csmr.ca.sandia.gov/~tgkolda/pubs/SAND2004-6391.pdf

[32] P. A. Telnoni, R. Budiawan, and M. Qana'a, "Comparison of machine learning classification method on text-based case in Twitter," in *Proc. International Conference on ICT for Smart Society*, 2019, pp. 1-5.

Ch A. S. Murty obtained a Master of Technology degree in JNT University, Hyderabad, India. He is currently pursuing PhD in Computer Science from National Forensic Sciences University, Gandhinagar, India. His research areas of interest are dark web, AI/ML Techniques, Auditing and assessment environment vulnerability assessment and penetration testing of infrastructure, web in IT/ ICT.

Dr. Parag H. Rughani obtained his PhD in computer science from Saurashtra University. He is currently working as an associate professor in digital forensics at the National Forensic Sciences University, India. His research thrust areas include machine learning, computer forensics, memory forensics and malware analysis.