INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

RESEARCH ARTICLE

Google Hacking Database Attributes Enrichment and Conversion to Enable the Application of Machine Learning Techniques

João Rafael Gonçalves Evangelista[1]*, **Renato José Sassi**[1], **Márcio Romero**[1]

1 Universidade Nove de Julho, São Paulo, PA, 01525-000, Brazil

OPEN ACCESS

Received: 18-07-2023

Accepted: 24-09-2023

Published: 13-11-2023

Citation: Evangelista JRG, Sassi RJ, Romero M (2023) Google Hacking Database Attributes Enrichment and Conversion to Enable the Application of Machine Learning Techniques . Indian Journal of Science and Technology 16(42): 3771-3777. https://doi.org/10.17485/IJST/v16i42.1799

* **Corresponding author**.

jrafael@uninove.edu.br

Funding: None

Competing Interests: None

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Objectives: Apply Natural Language Processing (NLP) to enrich Google Hacking Database (GHDB) with attributes and convert its textual values to ASCII, to enable the application of Machine Learning techniques to group Dorks by similarity and find vulnerabilities. **Methods:** The computational experiments were conducted in seven steps: Selection of the GHDB, Removal of Hyperlinks and Deletion of Attributes, Removal of the Site Parameter from Dorks, Removal of Outliers and Stopwords, Enrichment with NLP, Base Transformation, and Application of the Self-Organizing Maps (SOM). **Findings:** The application of NLP allowed segmenting of the Dorks by characters. After that, we converted the characters to their numeric values in ASCII. So, we enrich the GHDB and enable the application of ML techniques, in this case, the SOM. The results obtained with the application of the SOM were considered good. The topographic error (TE) and quantization error (QE) values of the maps generated by SOM were close to 0, which means good accuracy and the maps represent the input data well. **Novelty:** The formation of clusters of Dorks with SOM after enriching the GHDB with NLP.

Keywords: Google Hacking Database; Dorks; Natural Language Processing; SelfOrganizing Maps; Enrichment

1 Introduction

A method that can be used to ensure information security is to discover the vulnerabilities where the information is stored. Vulnerabilities are security flaws that pose risks to information. A practice used to find vulnerabilities in web pages is Google Hacking (GH). GH works like a google search that uses a search string, called Dork, which are sets of characters used to perform a specific search on Google[(1)].

Some authors, like[(2)] address types of information that can be found with GH. Information can be server names, open directories, file copies, IP address ranges, critical information about SCADA systems, online services, and devices such as cameras and printers. To assist the practice of GH, the Google Hacking Database (GHDB) is available on the internet, a base with Dorks evaluated and validated by offensive Security. Despite the number of Dorks, the GHDB contains few attributes, requiring that those who use

it have prior knowledge.

A disadvantage of the base is that it has few attributes: only the Dork, the author, the date of the publication and category. The Dorks available on the GHDB are classified into 14 categories, based on their functionality, that is, on the type of vulnerability they seek.

Furthermore, these few attributes of the base are textual values, not numeric, which limits its use of Machine Learning (ML) techniques, such as Artificial Neural Networks (ANNs). It is worth highlighting the importance of applying ML techniques in discovery of new information and knowledge about vulnerabilities.

ANNs are mathematical models of artificial intelligence inspired by the structure of the brain to simulate human behavior in processes such as learning, association, generalization, and abstraction. An ANN can learn and improve its performance based on the environment in which it finds itself. ANNs are very effective in solving nonlinear problems and performing parallel processing [3].

One type of ANN architecture that can be used in the Information Security area is the Kohonen Self-organizing Maps (SOM). According to [4], the SOM network is an ANN capable of extracting knowledge from a database, considering all its attributes simultaneously and forming clusters by similarity. The SOM is a network built around a one- or two-dimensional grid of neurons to capture the important characteristics contained in an input space (data) of interest. The SOM network is an ANN based on unsupervised learning capable of processing input from a multidimensional space, transforming it into a one-dimensional or two-dimensional array.

To assess the accuracy of the map and analyze whether the chosen topology is the one that "best represents the input vector data, some Accuracy measures can be used, such as the Quantization Error (QE) and the Topographic Error (TE).

The QE shows the quality of the input vector data. The QE will be close to zero when all nodes are well distributed in the map. The TE measures topology preservation of input data. As data is moving from multidimensional space to a two-dimensional or one-dimensional space, they end up losing information. One way to evaluate the representation of the initial input vector is using topographic error. When the topographic error is close to zero, it means that all nodes represent the initial input vector well.

This capability allows the SOM network to be applied in the Information Security area for various purposes, such as investigating digital evidence on computers. It is noteworthy that ML techniques have been used in Open-Source Intelligence practices (OSINT), such as the GH whose objective is to collect information from open sources [5].

So that ML techniques can be applied in GHDB it is necessary to enrich the base with attributes, to provide more information for the techniques to conduct their learning. Furthermore, it is also necessary to transform attributes with textual values into numeric values since artificial neural networks are mathematical models.

As for the GHDB enrichment, the Dorks can be divided by characters applying tokenization by Natural Language Processing (NLP). NLP is the subarea of Artificial Intelligence responsible for making computers able to interpret and develop content in human language. The application of NLP to texts or other human language source content can be performed through several tasks. Among the main tasks, the following stand out: stemming, corpus production, tokenization, lemmatization, grammatical marking, syntactic analysis, and the removal of stopwords [6].

Enrichment is the process responsible for adding information to a database, making it suitable for performing a certain task. When new information is added to a database, new facts are added to existing data, thus enabling new approaches to discover knowledge. As for the attributes with textual values of the GHDB, one way to transform them into numeric is to perform a character conversion to ASCII [7].

After the conversion of the textual value to numerical in ASCII, it will be possible to form groups of Dorks by similarities. That is, forming groups of Dorks by similar textual components, such as words and parameters.

In this scenario, the enrichment of a vulnerability databases to enable the application of ML techniques to extract new information was identified as a Research Gap in the literature, in this case, the Google Hacking Database. So, the objective of this paper was to apply NLP to enrich GHDB with attributes and convert its textual values to ASCII, to enable the application of ML techniques to group Dorks by similarity and find vulnerabilities.

The contributions of this paper are characterized by the description of how to apply NLP to enrich the GHDB, how to transform attributes with textual value or textual Dorks into numerical and how to apply an ANN, in this case, the SOM to group Dorks by similarity, enabling the application of an ML technique on such an important basis. These contributions are considered important for promoting the application of machine learning techniques on a relevant basis on vulnerabilities in websites and online applications.

2 Methodology

The literature review was performed using the following keywords: ”Natural Language Processing”, ”Google Hacking”, ”GHDB”, ”Dorks”, ”Artificial Neural Networks” in the databases: ACM Digital Library, EmeraldInsight, IeeeDigitalLibrary, and ScienceDirect.

The Dorks base selected was the GHDB (https://www.exploit-db.com/google-hacking-database) because the base has the largest number of documented and tested Dorks among all those available on the internet.

The GHDB has a total of 4,211 Dorks and 4 attributes, which are: Date: contains the date the Dork was published in the Base, Dork: contains the Dork and its access link, Category: informs which category the Dork belongs to, and Author: informs who sent Dork to the base.

So, the computational experiments were conducted in seven steps: Selection of the GHDB Base, Removal of Hyperlinks and Deletion of Attributes, Removal of the Site Parameter from Dorks, Removal of Outliers and Stopwords, Enrichment with Natural Language Processing, Base Transformation and Application of the SOM. Figure 1 presents the flowchart with the seven steps of computational experiments.

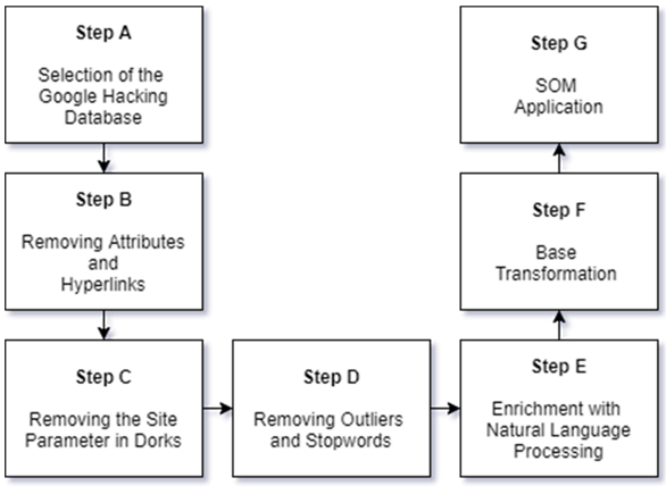

Fig 1. Seven Steps of Computational Experiments

a) Step A — Selection of the Google Hacking Database: In this step, the Dorks GHDB base was selected to conduct the computational experiments.

b) Step B — Removing Attributes and Hyperlinks: In this step, the hyperlinks embedded in Dorks were removed, along with nominal attributes from the GHDB database that were disregarded.

c) Step C — Removing the Site Parameter in Dorks: In this step, specific Dorks became in Dorks capable of running on any site. For this, the “Site” parameter present in Dorks was removed.

d) Step D — Removing Outliers and Stopwords: In this step, the removal of Outliers and Stopwords was conducted. Removed Stopwords were special characters present in Dorks. The removed Outliers were Composite Dorks and URLs.

e) Step E — Enrichment with Natural Language Processing: In this step, the base Dorks were selected and divided by characters applying tokenization by NLP. Then, the enrichment was conducted, transforming each Dork character into an attribute.

f) Step F — Base Transformation: In this step, the base Dorks were selected and converted to their respective numerical values in ASCII.

g) Step G — SOM Application: In this step, the SOM was applied to validate the GHDB enrichment and conversion, to generate similar Dorks clusters. Its performance will be evaluated by the Quantization Error (EQ) and Topographic Error (TE) values. Good results obtained in both errors will indicate whether the enriched and converted GHDB enabled the application of ML techniques.

3 Results and Discussion

The results of the computational experiments obtained with the application of the seven steps are presented below, shown in Figure 1.

a) Step A — Selection of the Google Hacking Database: At this step, Google Hacking Database (GHDB) from offensive Security was selected because it is an online base. It was necessary to copy the Dorks from the site and export them to a .csv file. The base has a total of 14 categories of Dorks. For this experiment, the Dorks of the categories: "Advisories and Vulnerabilities" and "Files Containing Juicy Info" were selected as a sample. These categories were chosen because they have the largest number of Dorks, respectively with 1996 and 450 Dorks.

b) Step B — Removing Attributes and Hyperlinks: In the Excel, the hyperlink from the Dorks and the author and date attributes from the base were removed, as these attributes do not influence the Dorks from the base. In this way, the base was left with 2 attributes remaining: Dork and Category.

c) Step C — Removing the Site Parameter in Dorks: Specific Dorks became Dorks capable of running on any site. For this purpose, the Site parameter was removed from the Dorks that had it. To remove the Site parameter, the Excel software was used and searched for the parameter: "Site:". After finding the Dorks that contained the "Site" parameter, these Dorks were modified, removing this parameter. Among the Dorks that had the "Site:" parameter, specific Dorks were found for Proxy Sites, Google Drive, Github, Mediafire, Dropbox, Sourceforge, and eBay.

d) Step D — Removing Outliers and Stopwords: At this step, when analyzing the Dorks, it was noticed that few had more than 100 characters in their composition. These Dorks had more than 100 characters for two main reasons: Composite Dorks and URLs. Thus, they were considered in this experiment as Outliers.

Composite Dorks are Dorks that have more than one Dork in their String. URLs are links to specific vulnerabilities on certain websites. Dorks that dealt with URLs were removed, as there would be no way to make them generic and thus automatically run them on other web pages. Composite Dorks were divided into smaller Dorks and then added to the base in their respective categories.

Then, the removal of Stopwords was performed to reduce noise at the base. This was necessary because in the GHDB database there are some Dorks with special characters that, when converted to their numerical value, have a value very different from the alphanumeric characters. To perform the removal of Stopwords, we defined 40 special characters as Stopwords to be removed. The removed Stopwords were as follows: **,';:;"'!?"()‘@~/|*[]^_.+**
%¨¬&©ºª}{£¢§-.

e) Step E — Enrichment with Natural Language Processing: In this step, the base Dorks were selected and divided by characters applying tokenization by NLP. It then made each character an attribute in the base. This was necessary because the base had only two attributes so far: Dork and Category. The low number of attributes makes it impossible to apply ML techniques on this basis.

To enrich this base, that is, add new attributes, an algorithm was developed in Python to discover the Dork with the greatest number of characters in its composition, and thus, create the same number of attributes in the Dorks Base. Thus, you can divide the Dork into characters and create new attributes in the base. This action not only enriches the base but also avoids in the next step of the experiment - F, when the Dork is converted to its numeric value in ASCII, that the numeric values obtained from the conversion are extensive, thus making impossible the application of ML techniques.

For example, a 10-character Dork, when converted to its numeric value, becomes a 30-digit numeric value. This is because each character converted to ASCII has a 3-digit numeric value. on the other hand, if each base attribute has only a single character, each attribute will receive a numeric value of 3 digits, enabling the application of intelligent techniques in the base. Thus, 94 attributes were created in the base, named Carac01, Carac02, Carac03 to Carac94. Thus, the database now has a total of 95 attributes, 94 "Carac" attributes added to the Category attribute with numerical values defined in steps C. The Dork division was performed through the "Nltk.word_tokenize()" function.

f) Step F — Base Transformation: After applying NLP in phase E, the Dorks characters were converted to their numerical values. For this, we selected the Dorks characters and converted them to their respective. To conduct this conversion, the study by [7] converts characters to their numeric ASCII value to detect Memory Overflow vulnerabilities. To conduct this conversion, the "ord" function of the Python language was used, the same function used in [7] study.

For example, the Dork: **inurl:/phpmyadmin/index.php?db=** in step D, this Dork was processed along with the other Dorks in the base, and thus, the special characters were removed. So, this Dork became: **inurlphpmyadminindexphpdb**.

Then, in step E, the Dorks were divided by characters, in this way, this Dork now has 25 characters, and that character was assigned to an attribute. In this way, the first character of this Dork: "i" was assigned to the attribute: Char01; the second character of this Dork: "n" was assigned to the attribute Char02 and so on until the end of the Dork. The other attributes received a value of 0 in order not to keep the base with null values. In this phase F, this Dork had its characters converted to its numeric

value in ASCII. Thus, the characters of this Dork now have the following value:

105 110 117 114 108 112 104 112 109 121 97 100 109 105 110 105 110 100 101 120 112 104 112 100 98.

g) Step G — SOM Application: After enriching and transforming the Dorks base, SOM was applied to validate the enrichment and conversion performed on the Dorks base, the possibility of applying ML techniques, and finding vulnerabilities in the generated clusters. For this, we sought to extract knowledge from the Dorks base with the application of SOM.

To perform the SOM, we defined the map dimension with 225 neurons, that is, a 15x15 map, and hexagonal topological neighborhood. In addition, the parameters used in the training phase were number of epochs (iterations) equal to 3000 and learning rate equal to 0.5.

For this experiment, all Dorks from the categories: "Advisories and Vulnerabilities" and "Files Containing Juicy Info" were selected as a sample. The two categories have the highest number of Dorks. The "Advisories and Vulnerabilities" category has a total of 1,996 Dorks, who search web pages with unprotected files. The application of SOM generated a map with 3 groups. This map is shown in Figure 2.

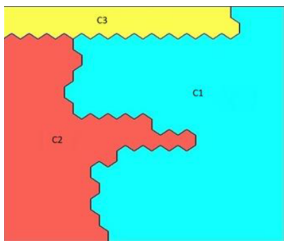

Fig 2. Map generated by the SOM network application in the "Advisories and Vulnerabilities"

The application of the SOM network generated three groups in the Advisories and Vulnerabilities base. Table 1 shows the characteristics of each one of them.

Table 1. Map Characteristics in the Advisories and Vulnerabilities Category

Cluster	Color	Dorks	Total (%)	Vulnerabilities
C1	Blue	1092	54,71%	Online Devices
C2	Red	622	31,16%	URL requests
C3	Yellow	282	14,13%	Multiple URL requests

It is observed in Table 1 that Dorks address vulnerabilities that allow advertisements and other messages on web pages. Such vulnerabilities seek online devices and URL requests to look for sensitive information.

The "Files Containing Juicy Info" category has a total of 450 Dorks that search for unprotected files with information about other systems on web pages. The application of SOM generated a map with 4 clusters. This map is shown in Figure 3.

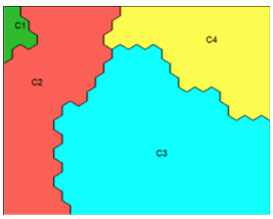

Fig 3. Map generated by the SOM network application in the "Files Containing Juicy Info"

The application of SOM generated four groups in the "Files Containing Juicy Info" base. Table 2 shows the characteristics of each one of them.

It is observed in Table 2 that Dorks address vulnerabilities that allow exploiting unprotected files with information about other systems on web pages. These vulnerabilities are of various extensions, spanning technologies such as SQL and Netscape.

Then, the accuracy of the map generated by the SOM was evaluated through Quantization Error (QE) and the Topographic Error (TE). Values are shown in Table 3

Table 2. Map Characteristics in the Files Containing Juicy Info Category

Cluster	Color	Dorks	Total (%)	Vulnerabilities
C1	Green	6	1,33%	Querys SQL Dump
C2	Red	135	30,00%	Email Servers
C3	Blue	228	50,67%'	Text Files and Github Files
C4	Yellow	81	18,00%	Files proj and Netscape

Table 3. Results of the metrics of the maps generated by the SOM network

Base	QE	TE
Advisories And Vulnerabilities	0,0990822	0.008849558
Files Containing Juicy Info	0,0202605	0.01703407

Analyzing the results in Table 3, the errors had values close to 0. This means that the topology of the input data was preserved, that is, that all nodes well represented the initial input vector. Thus, the errors obtained in the application of SOM can be considered as good.

Thus, it is understood that the enrichment of the GHDB database with NLP, together with the conversion of Dorks characters to numeric values in ASCII made it possible to apply an ML technique to generate similar Dorks groupings and to identify vulnerabilities. In this way, the procedure developed in this work made the GHDB base suitable for the application of ML techniques, such as ANNs. Procedures of this type are relevant because they enable the development of new research and, consequently, the discovery of new knowledge.

Enrichment tasks are commonly used, either with the combination of other databases, observing characteristics of existing data, or with techniques from different areas, like the [8] and [9]. Studies on database enrichment aim to add new attributes to a database, to improve the performance of Machine Learning techniques that will be applied to these databases.

Despite this, no other study was found in the literature that applied enrichment in the GHDB base to enable the application of Machine Learning techniques. Studies about GH are generally focused on the application in discovering vulnerabilities, such as the studies reported by [10] and [11].

4 Conclusion

This paper applied NLP to enrich the attributes of GHDB and convert its textual values into numeric values, using ASCII code to apply ML techniques. Therefore, the developed computational experiments included seven steps, which culminated in the validation by SOM of the GHBD enrichment and conversion, in addition to the generation of clusters with similar Dorks and the identification of vulnerabilities.

The results obtained with the application of the SOM were considered good, depending on the values presented by the metrics that evaluated the network. Thus, it is considered that the objective of this paper was achieved. With the base enriched and converted, it becomes possible to use other ML techniques to automate information security tests, such as in the construction of OSINT approaches or even for the creation of rules for defense systems such as Firewalls, IDS, and IPS, making them those capable of detecting GHDB practices.

It's important the developing research that seeks to extract knowledge or make it possible to extract knowledge from databases with data on vulnerabilities. This allows new solutions to be proposed and new studies to be developed with the objective of improving the security of web system users. Consequently, organizations benefit from using these safer systems, with a lower rate of vulnerabilities already known and stored in the databases, as is the case with GHDB. As for research on the practice of Google Hacking and the use of ML techniques in Dorks, no study on this topic was found. Thus, it is important to promote new studies that address the use of ML techniques in Dorks to combat cyber crime.

Among the limitations observed in this paper, the definition of stopwords stands out, because it does not find a pre-defined set of special characters, and the lack of studies in the literature to compare the results since the phases of conducting computational experiments was inspired by three different approaches.

The study conducted here does not intend to exhaust the subject, on the contrary, it sought to contribute to the Information Security area about the application of ML techniques in the identification of vulnerabilities when enriching and converting the GHBD. It is expected that the phases presented and applied in computational experiments can stimulate further research. This scenario, therefore, offers ample room for continuation work.

References

1) Kwak KT, Lee SY, Ham M, Lee SW. The effects of internet proliferation on search engine and over-the-top service markets. *Telecommunications Policy*. 2021;45(8):102146. Available from: https://doi.org/10.1016/j.telpol.2021.102146.
2) Mazurczyk W, Caviglione L. Cyber reconnaissance techniques. *Communications of the ACM*. 2021;64(3):86–95. Available from: https://doi.org/10.1145/3418293.
3) Bao W, Lianju N, Yue K. Integration of unsupervised and supervised machine learning algorithms for credit risk assessment. *Expert Systems with Applications*. 2019;128:301–315. Available from: https://doi.org/10.1016/j.eswa.2019.02.033.
4) Kohonen T. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*. 1982;43(1):59–69. Available from: https://doi.org/10.1007/BF00337288.
5) Evangelista JRG, Sassi RJ, Romero M, Napolitano D. Systematic Literature Review to Investigate the Application of Open Source Intelligence (OSINT) with Artificial Intelligence. *Journal of Applied Security Research*. 2021;16(3):345–369. Available from: https://doi.org/10.1080/19361610.2020.1761737.
6) Stahlberg F. Neural Machine Translation: A Review. *Journal of Artificial Intelligence Research*. 2020;69:343–418. Available from: https://doi.org/10.1613/jair.1.12007.
7) Guo H, Huang SG, Pan ZL, Hu JP, Hu ML. Research on Key Data Structure Localization Technology of Buffer Overflow Vulnerability. In: ICISS '18: Proceedings of the 1st International Conference on Information Science and Systems. ACM. 2018;p. 81–85. Available from: https://doi.org/10.1145/3209914.3226150.
8) Scheinert D, Casares F, Geldenhuys MK, Styp-Rekowski K, Kao O. Evaluation of Data Enrichment Methods for Distributed Stream Processing Systems. 2023. Available from: https://doi.org/10.48550/arXiv.2307.14287.
9) Platten JV, Sandels C, Jörgensson K, Karlsson V, Mangold M, Mjörnell K. Using Machine Learning to Enrich Building Databases—Methods for Tailored Energy Retrofits. *Energies*. 2020;13(10):1–22. Available from: https://doi.org/10.3390/en13102574.
10) Thomas H. Reconnaissance Techniques and Industrial Control System Tactics Knowledge Graph. In: Proceedings of the 22nd European Conference on Cyber Warfare and Security;vol. 22 (1). 2023;p. 688–695. Available from: https://doi.org/10.34190/eccws.22.1.1221.
11) Muhammad AB, Aminu Y, Sirina FI, Bello AI, Yusif M, Abubakar SMA, et al. Management of Vulnerabilities in Cyber Security. *Global Journal of Research in Engineering & Computer Sciences*. 2023;3(2):14–18. Available from: https://doi.org/10.5281/zenodo.7779507.