Malware Detection using Deep Neural Networks on Imbalanced Data

M. A. Mohammed[1*], D. A. Smait[2], M. Al-Tahai[3], I. S. Kamil[4], K. Al-Majdi[5] and Sh. K. Khaleel[6]
1- Department of Anesthesia Techniques, Al-Noor University College, Bartella, Iraq.
Email: mohammed.abdulkreem@alnoor.edu.iq (Corresponding author)
2- The University of Mashreq, Iraq.
Email: draialisawi@uom.edu.iq
3- Medical technical college/ Al-Farahidi University, Baghdad, Iraq
Email: m.altahi6010@gmail.com
4- Medical Laboratories Techniques Department, Al-Mustaqbal University College, Babylon, Iraq.
Email: israasaleh@mustaqbal-college.edu.iq
5- Department of biomedialc engineering, Ashur University College, Baghdad, Iraq.
Email: dr.kadhum@au.edu.iq
6- Al-Esraa University College, Baghdad, Iraq.
Email: shahd@esraa.edu.iq

Received: 2 October 2022 Revised: 28 October 2022 Accepted: 13 November 2022

ABSTRACT:
Through the use of malware, particularly JavaScript, cybercriminals have turned online applications into one of their main targets for impersonation. Detection of such dangerous code in real-time, therefore, becomes crucial in order to prevent any harmful action. By categorizing the salient characteristics of the malicious code, this study suggests an effective technique for identifying malicious Java scripts that were previously unknown, employing an interceptor on the client side. By employing the wrapper approach for dimensionality reduction, a feature subset was generated. In this paper, we propose an approach for handling the malware detection task in imbalanced data situations. Our approach utilizes two main imbalanced solutions namely, Synthetic Minority Over Sampling Technique (SMOTE) and Tomek Links with the object of augmenting the data and then applying a Deep Neural Network (DNN) for classifying the scripts. The conducted experiments demonstrate the efficient performance of our approach and it achieves an accuracy of 94.00%.

KEYWORDS: Malware Detection, Imbalanced Data, Convolutional Neural Networks, SMOTE, Tomek Links

1. INTRODUCTION

Web applications that are accessed through web browsers have evolved into a top target for cybercriminals as a result of the quick development of contemporary computers and network infrastructure, as well as our growing reliance on the Internet and its services [1]. According to Symantec research published in 2016, there were found to be 430 million distinct malware samples, up 36% from 2014 [2]. Cybercriminals target people and businesses by using harmful programs and malicious URLs. By harming or impairing computer and system functionality, carrying out phishing assaults, showing intrusive adverts, and extorting money, these attacks serve only to further personal, monetary, and political goals. Therefore, one of the biggest security challenges is detecting such malicious code assaults [3].

Cross-site scripting (XSS), one of the top ten vulnerabilities, is regarded as the second most serious vulnerability by the Open Web Application Security Project (OWASP) [4]. XSS now accounts for 43% of all vulnerabilities that have been disclosed. A sort of injection attack known as XSS involves inserting malicious scripts around safe code on a trustworthy website in order to acquire cookies, sessions, and other sensitive data [5].

The HTTP protocol, client- and server-side applications, web browsers, and scripting methods like javascript and CGI are just a few of the many mechanisms and technologies that make up the web system [6]. Because of the inconsistencies in these technologies, applications built on them are extensively used, but at the expense of their security. As a result, it is difficult to build security on these infrastructures, and

Paper type: Research paper
DOI: 10.30486/mjee.2022.696523
How to cite this paper: M. A. Mohammed, D. A. Smait, M. Al-Tahai, I. S. Kamil, K. Al-Majdi and Sh. K. Khaleel, "Malware Detection using Deep Neural Networks on Imbalanced Data", Majlesi Journal of Electrical Engineering, Vol. 16, No. 4, pp. 123-129, 2022.

many web apps built on them are vulnerable to security flaws [7].

Web-based apps are often used as a target and one of the primary ways to breach the integrity of the system and networks due to their public accessibility. Additionally, due to their widespread installation, web services and host systems (such as virtual environments, PCs, servers, IoT devices, and ICS) are attractive targets for computer and ICS malware that spreads itself across internal and external networks by exploiting web-related security problems [8]. This is true for a number of settings, including open websites, business networks, and ICS/SCADA systems that store command and control information in their databases and, as a result, where access via a web browser might potentially result in the attacker taking complete access to the system [9].

The code segment or a file in sandbox machine learning has been introduced quickly, accurately, and without isolation to assess if the code or a file is suspicious or not. With a very low false-positive rate and the capacity to learn from patterns, machine learning is able to identify new malware variants [10]. Machine learning is quite useful, especially given the malware's increasingly versatile nature. In particular, machine learning-based malware detection methods can result in quick malware detection and do not necessitate routine client-side anti-malware program updates.

The new field of study known as deep learning, a subclass of machine learning, is focused on the structure and function of humans and imitates how the human brain functions using neural networks. In a variety of applications, including speech recognition [11], image classification [12], medical image analysis [13], and engineering [14]-[15]-[16], deep learning has propelled and proven successful. The development of neural networks with the goal of overcoming the drawbacks of earlier machine learning techniques such as naive Bayes, hidden Markov models, and support vector machines. Neural networks may therefore produce substantially better categorization accuracy. Building a deep learning neural network, also known as a neural network with more prospect layers, has the potential to increase classification accuracy [17].

The data imbalance in machine learning and deep learning makes it difficult to undertake data analytics in practically every field of real-world study. As in the cases of computer vision, marketing, and information security the raw primary data frequently suffers from the skewed perspective of data distribution of one class over the other [18]. An example of a classification issue where the distribution of instances among the recognized classes is biased or unbalanced is an imbalanced classification problem. One case in the minority class for hundreds, thousands, or millions of examples in the majority class or classes might indicate a mild bias all the way up to a serious imbalance. Predictive modeling is challenged by imbalanced classifications since the majority of machine learning methods for classification were built on the premise that there should be an equal number of samples in each class. As a result, models perform poorly in terms of prediction, particularly for the minority class [19].

In this paper, we aim to present a deep learning-based approach for addressing the problem of malware detection from source codes. Our approach is designed to function effectively in situations where the data distribution is imbalanced. Our approach uses Synthetic Minority Over Sampling Technique (SMOTE) [20] and Tomek Links [21] technique to augment the training data and this leads to a better training process. Then a deep neural network acts as a binary classifier for classifying benign from malicious scripts. The conducted and extensive experiments on a collected dataset from real IT-based companies located in Iraq demonstrate the efficacy of the proposed pipeline. Overall, our main contributions in this paper are itemized as the following:

1. A deep learning-based approach is proposed for detecting malware from source scripts.
2. Our approach is robust enough since it uses resampling techniques for better handling the imbalanced data.
3. We compare our results with machine-learning-based algorithms and show the superiority of the proposed model.
4. A new dataset is collected from real IT-based companies located in Iraq.

2. MATERIALS AND METHODS

2.1. Dataset

We used well-known JavaScript malware sources like HynekPetrak/javascript-malware-collection on GitHub to obtain harmful programs. First, during the code gathering stage, we employed the repository's code, where the existence or absence of malware is predefined, as the main source of information for this study. We took the code out of the repository, marked it as harmful code, and put it away for later use. Additionally, we took the innocuous example code files from popular packages that thousands of developers use from various GitHub repositories. They served as an example of sound coding that was given a benign title.

In order to simulate a riddle, the mathematical field of graph theory was initially created in the 18th century. Graphs are excellent for building straightforward, abstract models. Mathematicians and scientists may include a variety of well-known ideas, techniques, and theories into their models thanks to graph theory [22]. A graph is a collection of vertex and vertex pairs. Following the principles of Graph theory, in this work, we extract Control Flow Graph (CFG) from the scripts with the aim of, further, extracting features for the model. Further, the distribution of the collected data is

depicted in Table 1.

Table 1. The distribution of data.

Class	Training	Validation	Testing
Malicious	1000	500	500
Benign	4000	500	2500

2.2. Convolutional Neural Network

Deep learning is a particular branch of machine learning that emphasizes the learning of successive layers of more meaningful representations [23]. The deep in deep learning refers to multiple layers of representations rather than any form of deeper knowledge that may be attained through the method. The depth of the model refers to the number of layers that go into a data model. Layered representations learning and hierarchical representations learning could have been better titles for the discipline. Today's deep learning techniques frequently comprise tens or even hundreds of representational layers that are learned automatically through exposure to training data [24].

A Convolutional Neural Network (CNN) is a deep learning system that can take in an input image, give various elements and objects in the image importance (learnable weights and biases), and be able to distinguish between them [25]. Comparatively speaking, a CNN requires substantially less pre-processing than other classification methods. CNN can learn these filters/characteristics with adequate training, but in primitive approaches filters are hand-engineered. The structure of a CNN was influenced by the way the visual cortex is organized and is similar to the connection pattern of neurons in the human brain. Only in this constrained area of the visual field, known as the Receptive Field, do individual neurons react to stimuli [26].

Through the use of pertinent filters, a CNN may effectively capture the spatial and temporal relationships in the input data. Because there are fewer factors to consider and the weights may be reused, the architecture provides a better fitting to the picture dataset. In other words, the network may be trained to better comprehend the level of complexity in the data [27].

The parameters of the convolution layer are made up of a collection of K learnable filters, or "kernels," each of which has a width and a height and is almost usually square. Although these filters are modest (in terms of their spatial dimensions), they cover the whole volume's depth. The depth is the number of channels in the data for inputs to CNN (i.e., a depth of three when working with RGB images, one for each channel) [28]. The number of filters used in the preceding layer will determine the depth for volumes farther down the network. We now obtain K, 2-dimensional activation maps after applying all K filters to the input volume. The final output volume is created by stacking our K activation maps along the depth dimension of our array. Thus, each item in the output volume represents the result of a neuron that only "looks" at a portion of the input [29]. By doing this, the network "learns" filters that turn on when certain kinds of features are present at a certain spatial region within the input volume. In CNNs, the idea of convolving a tiny filter with a large(r) input volume has particular significance for the local connectivity and receptive field of a neuron [30].

2.3. Synthetic Minority Over Sampling Technique

In the context of learning from imbalanced data, the SMOTE preprocessing procedure is regarded as a "de facto" standard [31]. This is a result of the procedure's straightforward design and versatility when used to solve various kinds of issues. SMOTE has demonstrated effectiveness in several applications across numerous fields since its release in 2002. SMOTE has substantially influenced new supervised learning paradigms, such as multilabel classification, incremental learning, semi-supervised learning, and multi-instance learning, among others [32]. It has also inspired a number of strategies to address the problem of class imbalance. SMOTE is used to balance datasets with a markedly uneven ratio and it seeks to increase the number of minority class samples by producing synthetic samples in the minority class. The synthetic production of fresh samples is different from the multiplication technique in that it avoids the overfitting problem. By interpolating between samples of this class that are in close proximity to one another, SMOTE's principal goal is to create fresh samples of data in the minority class. SMOTE thereby boosts the proportion of minority class examples in an unbalanced dataset, which helps the classifier attain more generalizability [33].

2.4. Tomek Links

Classification modeling frequently runs into the issue of an unbalanced dataset in real-world applications, where the majority class has a substantially larger number of members than the minority class, making it difficult for the model to effectively learn from the minority class. When data from the minority class is increasingly crucial, such as in illness diagnosis datasets, churn datasets, and fraud detection datasets, this poses a major dilemma [34].

Oversampling the minority class or undersampling the majority class are two common solutions to the imbalance dataset problem. However, each of these strategies has a flaw. The goal behind the traditional oversampling approach is to replicate a few random instances from the minority class. As a result, this method does not extract any fresh information from the data. Contrarily, the undersampling strategy involves deleting a few random samples from the majority class at the expense of also losing some information from the

original data [35]. Another effective undersampling method for rebalancing the data is Tomek Links (TL). If the following inequalities are met for a given pair of samples (Ai, Aj) from the datasets, we may declare that they constitute a TL:

$$\boldsymbol{d(A_i, A_l) < d(A_i, A_j)\ or\ d(A_j, A_l) < d(A_i, A_j)}$$

Where, d(x,y) is the distance between x and y.

3. RESULTS AND DISCUSSION

3.1. Experimental Setup

By employing assessment measures like accuracy, true positive rate, and false positive rate, performance evaluation aims to research and examine how well classifiers function in accurately identifying the instance. Our classifier underwent a 5-fold cross-validation evaluation. Predictive models are evaluated using the K-fold cross-validation technique, which divides the original sample into a training set and a test set. The data is divided into 5 subgroups for 5-fold cross-validation, with the final subset serving as the test set. The remaining nine subsets are utilized to train the classifier.

The proposed classifier's architecture is depicted in Table 2.

Table 2. The proposed classifier architecture.

Number	Type	Dim
1	Convolutional	128
2	ReLU	-
3	Convolutional	64
4	ReLU	-
5	Convolutional	32
	Pool	(2,2)
	Fully Connected	16

3.2. Evaluation M etrics

Performance evaluation serves as a versatile technique that is used to compare the system's actual values to the predicted values. Our evaluation's goal is to analyze and gauge a classifier's effectiveness in spotting dangerous code. We are really concerned about accuracy, which is defined as equation 1, in order to get high outcomes from the suggested technique.

$$Accuracy = \frac{Number\ of\ correctly\ classified\ samples}{Total\ number\ of\ samples} \quad (1)$$

When the attack detection method wrongly perceives a legitimate code as harmful code, a false positive situation happens. A false negative happens in a particular method when harmful code is not found while engaging in prohibited behavior. The assessment of false positives and false negatives is done using a confusion matrix or error matrix to determine the detection rate. Equations 2 and 3 determine the false positive and false negative detection rates.

$$FPR = \frac{FP}{FP+TN} \quad (2)$$

$$FPR = \frac{FN}{FN+TP} \quad (3)$$

Where FN stands for false negative, TN for true negative and TP for true positive, and FPR stands for false positive rate.

True positive demonstrates that some harmful samples have been successfully recognized, whereas false positive suggests that some negative samples have been mistakenly labeled as malicious. True negative demonstrates that some negative samples have been accurately identified. The speed at which the malicious scripts are handled will determine how well the suggested detection method performs. The system resource consumption in both circumstances will be used to determine the performance by measuring the latency time required to show a page in both the existence and lack of an interceptor. Where the detecting system is inadequate, real-time detection cannot be achieved. The evaluation procedure has also made use of the receiver operating characteristic (ROC). For a binary classifier with different thresholds, the ROC is a graphical representation created by comparing the percentage of TPR versus the fraction of FPR.

3.3. Classification

In this section, the results of classification are demonstrated. Table 3 shows the achieved results of the 5-fold experiment.

Table 3. The distribution of data

Fold	Accuracy	Precision	Recall	F1-Score
1	94.20	96.00	92.66	94.30
2	94.15	96.21	92.73	94.34
3	93.90	95.80	91.67	93.92
4	95.10	96.11	92.80	94.41
5	94.32	96.10	92.43	94.58

Moreover, Figs. 1 to 5 demonstrate the achieved confusion matrix for folds number 1 to 5, respectively.

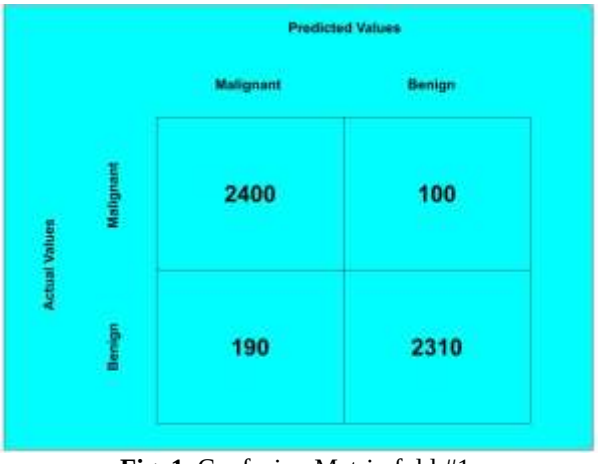

Fig. 1. Confusion Matrix fold #1.

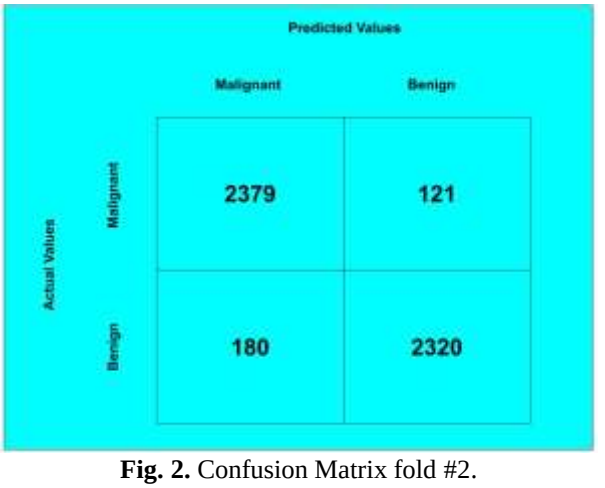

Fig. 2. Confusion Matrix fold #2.

/

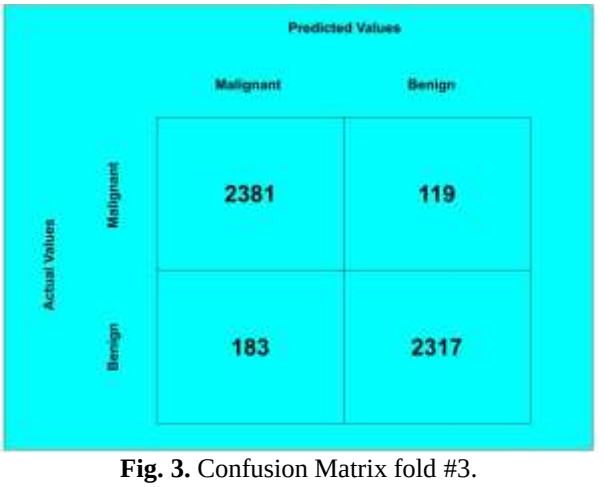

Fig. 3. Confusion Matrix fold #3.

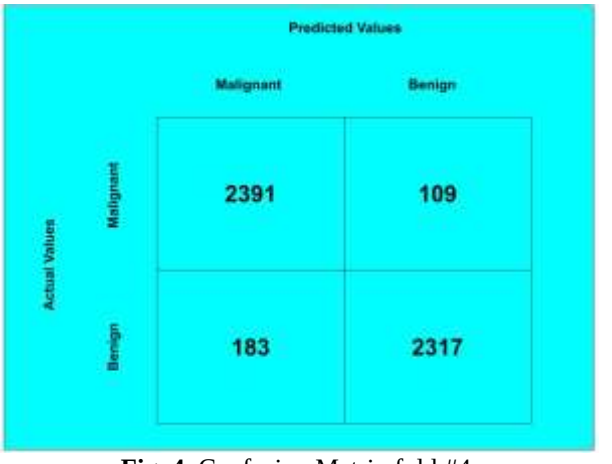

Fig. 4. Confusion Matrix fold #4.

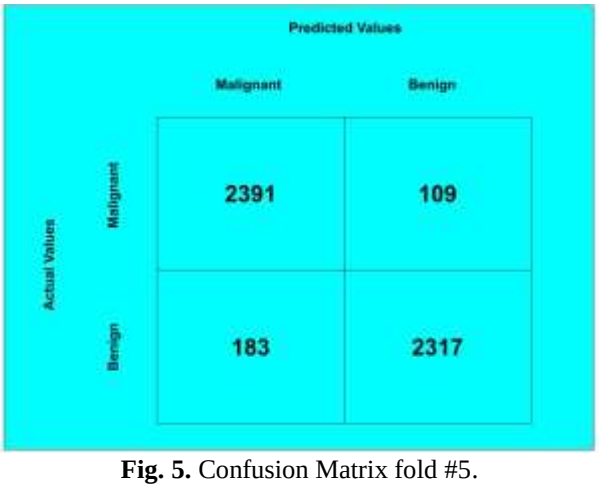

Fig. 5. Confusion Matrix fold #5.

Further, we have compared our proposed methodology with machine learning-based algorithms, namely Support Vector Machine (SVM), Random Forest (RF), and Decision Tree (DT). In fact, these algorithms play as the baseline for comparing the efficacy of the proposed model. Fig. 6 shows this comparison in terms of accuracy, precision, and recall.

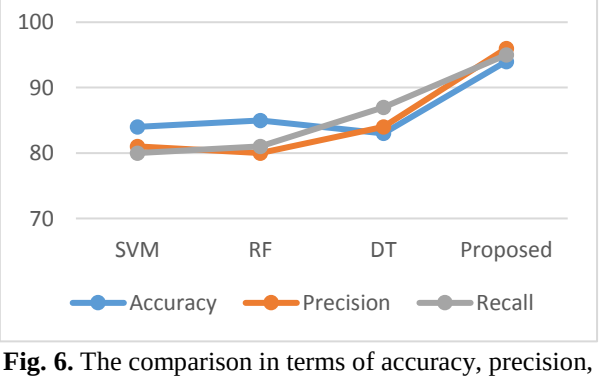

Fig. 6. The comparison in terms of accuracy, precision, and recall.

As is seen in Fig. 6, the proposed model possesses superior performance with respect to various aspects. First of all, the high accuracy proves the better performance of the model in classifying the correct samples. On the other hand, high recall claims the excellency of the proposed model in terms of fetching the relevant data for both benign and malicious samples. This shows the reliability of the model in imbalanced situations. High precision shows the good ability of the classifier for detecting benign samples.

4. CONCLUSION

Cybercriminals have made the web application one of their primary targets for impersonation by using malware, notably JavaScript. Therefore, it becomes essential to identify such unsafe code in real-time in order to stop any negative actions. This study provides a useful method for recognizing malicious Java scripts that were previously unknown, using an interceptor on the client side, by classifying the key traits of the malicious code. A feature subset was produced using the wrapper technique for dimensionality reduction. In this research, we suggest a method for addressing malware detection tasks when the data is unbalanced. Our approach is validated by conducting extensive experiments on a collected dataset from real IT-based companies. The experiments show a great performance of the proposed pipeline in detecting malicious scripts.

REFERENCES

[1] Rai, Mahima, and Hardwari Mandoria. **"A study on cyber crimes cyber criminals and major security breaches."** *Int. Res. J. Eng. Technol*, Vol.6, No. 7, pp.1-8, 2019.

[2] Gursoy, Mehmet Emre, Acar Tamersoy, Stacey Truex, Wenqi Wei, and Ling Liu. **"Secure and utility-aware data collection with condensed local differential privacy."** *IEEE Transactions on Dependable and Secure Computing*, Vol. 18, No. 5, pp. 2365-2378, 2019.

[3] Fang, Yong, Cheng Huang, Yu Su, and Yaoyao Qiu. **"Detecting malicious JavaScript code based on semantic analysis."** *Computers & Security*, Vol. 93, p. 101764, 2020.

[4] Rodríguez, Germán E., Jenny G. Torres, Pamela Flores, and Diego E. Benavides. **"Cross-site scripting (XSS) attacks and mitigation: A survey."** *Computer Networks*, Vol. 166, p. 106960, 2020.

[5] Tariq, Iram, Muddassar Azam Sindhu, Rabeeh Ayaz Abbasi, Akmal Saeed Khattak, Onaiza Maqbool, and Ghazanfar Farooq Siddiqui. **"Resolving cross-site scripting attacks through genetic algorithm and reinforcement learning."** *Expert Systems with Applications* Vol. 168, p. 114386, 2021.

[6] Mokbal, Fawaz Mahiuob Mohammed, Wang Dan, Wang Xiaoxi, Zhao Wenbin, and Fu Lihua. **"XGBXSS: an extreme gradient boosting detection framework for cross-site scripting attacks based on hybrid feature selection approach and parameters optimization."** *Journal of Information Security and Applications*, Vol. 58, p. 102813, 2021.

[7] Odun-Ayo, Isaac, Williams Toro-Abasi, Marion Adebiyi, and Oladapo Alagbe. "**An implementation of real-time detection of cross-site scripting attacks on cloud-based web applications using deep learning.**" *Bulletin of Electrical Engineering and Informatics*, Vol. 10, No. 5, pp. 2442-2453, 2021.

[8] Usha, G., S. Kannimuthu, P. D. Mahendiran, Anusha Kadambari Shanker, and Deepti Venugopal. **"Static analysis method for detecting cross site scripting vulnerabilities."** *International Journal of Information and Computer Security*, Vol. 13, No. 1, pp. 32-47, 2020.

[9] Kadhim, R., and M. Gaata. **"A hybrid of CNN and LSTM methods for securing web application against cross-site scripting attack."** *Indones. J. Electr. Eng. Comput. Sci*, Vol. 21, pp. 1022-1029, 2020.

[10] Alsaffar, Mohammad, Saud Aljaloud, Badiea Abdulkarem Mohammed, Zeyad Ghaleb Al-Mekhlafi, Tariq S. Almurayziq, Gharbi Alshammari, and Abdullah Alshammari. **"Detection of Web Cross-Site Scripting (XSS) Attacks."** *Electronics*, Vol. 11, No. 14, p. 2212, 2022.

[11] Nassif, Ali Bou, Ismail Shahin, Imtinan Attili, Mohammad Azzeh, and Khaled Shaalan. **"Speech recognition using deep neural networks: A systematic review."** *IEEE access,* Vol. 7, pp. 19143-19165, 2019.

[12] Obaid, Kavi B., Subhi Zeebaree, and Omar M. Ahmed. **"Deep learning models based on image classification: a review."** *International Journal of Science and Business*, Vol. 4, No. 11, pp. 75-81, 2020.

[13] Karimi, Davood, Haoran Dou, Simon K. Warfield, and Ali Gholipour. **"Deep learning with noisy labels: Exploring techniques and remedies in medical image analysis."** *Medical Image Analysis*, Vol. 65, p. 101759, 2020.

[14] Nourani, Vahid, Zahra Razzaghzadeh, Aida Hosseini Baghanam, and Amir Molajou. **"ANN-based statistical downscaling of climatic parameters using decision tree predictor screening method."** *Theoretical and Applied Climatology*, Vol. 137, No. 3, pp. 1729-1746, 2019.

[15] Sharghi, Elnaz, Vahid Nourani, Hessam Najafi, and Amir Molajou. **"Emotional ANN (EANN) and wavelet-ANN (WANN) approaches for Markovian and seasonal based modeling of rainfall-runoff process."** *Water resources management*, Vol. 32, No. 10, pp. 3441-3456, 2018.

[16] Nourani, Vahid, Amir Molajou, Selin Uzelaltinbulat, and Fahreddin Sadikoglu. **"Emotional artificial neural networks (EANNs) for multi-step ahead prediction of monthly precipitation; case study: northern Cyprus."** *Theoretical and Applied Climatology*, Vol. 138, No. 3, pp. 1419-1434, 2019.

[17] Jia, Sen, Shuguo Jiang, Zhijie Lin, Nanying Li, Meng Xu, and Shiqi Yu. **"A survey: Deep learning for hyperspectral image classification with few labeled samples."** *Neurocomputing*, Vol. 448, pp. 179-204, 2021.

[18] Kaur, Harsurinder, Husanbir Singh Pannu, and Avleen Kaur Malhi. "**A systematic review on imbalanced**

data challenges in machine learning: Applications and solutions." *ACM Computing Surveys (CSUR)*, Vol. 52, No. 4, pp. 1-36, 2019.

[19] Zhang, Wei, Xiang Li, Xiao-Dong Jia, Hui Ma, Zhong Luo, and Xu Li. **"Machinery fault diagnosis with imbalanced data using deep generative adversarial networks."** *Measurement*, Vol. 152, p. 107377, 2020.

[20] Ishaq, Abid, Saima Sadiq, Muhammad Umer, Saleem Ullah, Seyedali Mirjalili, Vaibhav Rupapara, and Michele Nappi. **"Improving the prediction of heart failure patients' survival using SMOTE and effective data mining techniques."** *IEEE access*, Vol. 9, pp. 39707-39716, 2021.

[21] Swana, Elsie Fezeka, Wesley Doorsamy, and Pitshou Bokoro. **"Tomek Link and SMOTE Approaches for Machine Fault Classification with an Imbalanced Dataset."** *Sensors*, Vol. 22, No. 9, pp. 3246, 2022.

[22] Farahani, Farzad V., Waldemar Karwowski, and Nichole R. Lighthall. **"Application of graph theory for identifying connectivity patterns in human brain networks: a systematic review."** *frontiers in Neuroscience*, Vol. 13, p. 585, 2019.

[23] Khan, Asifullah, Anabia Sohail, Umme Zahoora, and Aqsa Saeed Qureshi. **"A survey of the recent architectures of deep convolutional neural networks."** *Artificial intelligence review*, Vol. 53, No. 8, pp. 5455-5516, 2020.

[24] Kattenborn, Teja, Jens Leitloff, Felix Schiefer, and Stefan Hinz. **"Review on Convolutional Neural Networks (CNN) in vegetation remote sensing."** *ISPRS Journal of Photogrammetry and Remote Sensing*, Vol. 173, pp. 24-49, 2021.

[25] Capra, Maurizio, Beatrice Bussolino, Alberto Marchisio, Muhammad Shafique, Guido Masera, and Maurizio Martina. **"An updated survey of efficient hardware architectures for accelerating deep convolutional neural networks."** *Future Internet*, Vol. 12, No. 7, p. 113, 2020.

[26] Boulent, Justine, Samuel Foucher, Jérôme Théau, and Pierre-Luc St-Charles. **"Convolutional neural networks for the automatic identification of plant diseases."** *Frontiers in plant science*, Vol. 10, p. 941, 2019.

[27] Véstias, Mário P. **"A survey of convolutional neural networks on edge with reconfigurable computing."** *Algorithms*, Vol. 12, No. 8, p. 154, 2019.

[28] Jiao, Jinyang, Ming Zhao, Jing Lin, and Kaixuan Liang. **"A comprehensive review on convolutional neural network in machine fault diagnosis."** *Neurocomputing*, Vol. 417, pp. 36-63, 2020.

[29] Ribani, Ricardo, and Mauricio Marengoni. **"A survey of transfer learning for convolutional neural networks."** In *2019 32nd SIBGRAPI conference on graphics, patterns and images tutorials (SIBGRAPI-T)*, pp. 47-57. IEEE, 2019.

[30] Tulbure, Andrei-Alexandru, Adrian-Alexandru Tulbure, and Eva-Henrietta Dulf. **"A review on modern defect detection models using DCNNs–Deep convolutional neural networks."** *Journal of Advanced Research*, Vol. 35, 33-48, 2022.

[31] Kovács, György. **"Smote-variants: A python implementation of 85 minority oversampling techniques."** *Neurocomputing*, Vol. 366, pp. 352-354, 2019.

[32] Zhang, Hongpo, Lulu Huang, Chase Q. Wu, and Zhanbo Li. **"An effective convolutional neural network based on SMOTE and Gaussian mixture model for intrusion detection in imbalanced dataset."** *Computer Networks*, Vol. 177, p. 107315, 2020.

[33] Raghuwanshi, Bhagat Singh, and Sanyam Shukla. **"SMOTE based class-specific extreme learning machine for imbalanced learning."** *Knowledge-Based Systems* Vol. 187, p. 104814, 2020.

[34] Kumar, Sujit, Saroj Kr Biswas, and Debashree Devi. **"TLUSBoost algorithm: a boosting solution for class imbalance problem."** *Soft Computing*, Vol. 23, No. 21, pp. 10755-10767, 2019.

[35] Devi, Debashree, and Biswajit Purkayastha. **"Redundancy-driven modified Tomek-link based undersampling: A solution to class imbalance."** *Pattern Recognition Letters*, Vol. 93, pp. 3-12, 2017.