Automated Malware Classification based on Hybrid CNN-SVM framework

Ishan Jain, Srishti Vashishtha, Deepika Varshney, Rachna Narula, Vinayak Sharma

[1,5] Student, [2-4]Assistant Professor
[1,2,4,5] Department of Computer Science & Engineering
[1,2,4,5] Bharati Vidyapeeth's College of Engineering
New Delhi, India
[3]CSE & IT department,
[3]Jaypee Institute of Information Technology, Noida, India

Abstract - Research is being conducted on malware analysis and classification using various models and techniques. The increasing presence of malware code files has made manual analysis time-consuming, so efficient tools are needed to quickly detect malware. One popular technique is using machine learning models to classify malware code as images, which simplifies the detection process. The objective is to train a model that can classify new malware files on its own, using techniques such as CNNs for image processing and subsequent classification. This paper proposes the usage of a CNN + SVM model for classification which is shown to outperform popular classification methodologies.

Index Terms - Malware analysis, Malware classification, Machine learning, Image processing, Neural networks

I. INTRODUCTION

In the modern world, malware attacks are fairly widespread and pose serious threats to a variety of industries every day. Malware detection and analysis have seen immense research in the past decade because of the wide variety of obfuscation techniques and the rapid pace at which malware families have evolved and become more dangerous. There have been several new variants of malware that have been coming up, and one of the primary steps in understanding and dealing with malicious code is to be able to carry out their successful classification. Researchers have been facing problems based on the usage of polymorphic malware by hackers to invade detection by traditional methods.

Malware attacks are becoming more common, and malware detection and analysis is a complex and rapidly-evolving field. In recent years, there's been a rise in new variants of malware, as well as new ways malware can be disguised. To complicate matters, researchers are also facing new challenges, such as malware that uses encryption and packers.

II. LITERATURE SURVEY

The upcoming section contains a literature review cor- responding to the domain of our problem statement. Conventional methods of malware detection include static and dynamic analysis [1]. Regardless of their dependability traditionally, the techniques are less efficient than artificial intelligence-based techniques owing to their lack of scalability [2], especially in dynamic analysis where the usage of a virtual environment serves as an added disadvantage. Machine Learning (ML) based techniques allow the visibility of similar patterns between malware families and their interaction with one another [3].

The past 10-15 years have been full of a lot of work in building intelligent ML-based malware classification models. These use a variety of techniques used in machine learning, namely Support vector Machines (SVM) [4], Naive Bayes classifiers [5], multidimensional classifiers [6], and so on, but success has been found with the usage of modern ML techniques such as Convolutional Neural Networks (CNN) and K-Nearest Neighbours (KNN) [7]. The usual issue faced by a CNN is large sample sizes of malware, and great modifications of the model are required to rectify the problems of detection and classification. Therefore, the deeper architecture of CNNs is required for malware classification [8]. In some cases, classification methods and approaches have required the execution of actual malware code as well, while in some other related research, the extent of viewing malware families as digital images and classifying them into families is limited [9].

Recurrent neural networks (RNN) are often used in the feature extraction step of malware analysis [10]. Feature extraction is an essential step in the training of the classifier for malware detection. Along with methods to reduce the dimensionality of input data

into the ML model, researchers use RNNs in modern methods. Subsequently, the need arises to improve the success rate of the model, hence neural nets are applied as additional layers, as done by Dahl et. al [11]. Other approaches used in the past are as follows: (1) analysis of the metadata of import and export tables and (2) clustering techniques to build graph-based methods [12].

However, among all the existing research in malware classification, neural network-based approaches, especially those built upon CNN have been found to be most successful to build a generic framework for the purpose of classifying malware samples into their respective families [13]. Building upon the work of Kalash et. al [14], Malimg, and Microsoft datasets, we have used an upgraded and refined CNN model for malware family classification to build upon pre-existing approaches. K- Nearest Neighbor-based classification techniques are also implemented to compare the accuracy and other important metrics of the success of the technique.

III. BACKGROUND

Classification problems of different areas [22-25] can be solved by applying various machine learning based algorithms. The understanding and working of these ML algorithms is a pre-requisite for conducting research. The following section provides the background.

A. Malware Analysis

Malware refers to programs or software that are written with the intention of causing harm to a user, system, or network. Malware are also of various types and families, such as trojans, spyware, bots, worms, viruses, etc. With the advent of digital technology, the number and complexity of malicious software have increased multi-fold in the past few years. Malware analysis refers to understanding the behavior of suspicious files and then developing methods to identify such files. Malware analysis techniques can be static, dynamic, and hybrid.

Static malware analysis is done by looking at the malware's programmed code to gain a better insight into the working of the malware. Antivirus software will be executed on the malware during static malware analysis. Shell scripts and other files are inspected. Examples are debuggers, disassemblers, and decompilers. The IT staff or relevant authorities would be able to create stronger safeguards by understanding how the code works. In static analysis, static examination of aspects such as IP addresses, file names, hashes, etc. is used to determine the malicious nature of the file.

Dynamic malware analysis is a quicker malware analysis approach. The relevant authorities examine how the malware continues to operate when doing dynamic malware analysis.

For a baseline system, one sees what changes the virus does. It is necessary that the malware lab is not linked to any other network and read-only media must be used to transmit files. There are several system modifications that should be a red flag. One must check to see if any new services have been deployed if any system settings have been changed, and if there are any new processes running. Dynamic analysis involves executing the suspected file in a sandbox to provide deeper visibility and insight.

A hybrid analysis is a mix of the better aspects of both approaches to extract many more indicators of compromise (IOC). It uses strategies from both methodologies to com- pensate for each other's flaws. When unpacking binary files or reading them in assembly code, some operations that are concealed at run-time may be recognized. Similarly, when an obfuscated opcode executes and the actions or outcomes are identified in real-time, the obfuscated opcode may be disclosed.

B. Neural Networks

A neural network is a machine learning structure that mim- ics the human brain, and information transfer is represented similarly to how electric signals move between neurons. The artificial neurons are arranged in layers and data travels from the input layers to the final output layer [21]. In a general neural network, the layers are as follows:

- Input Layer - It is the first layer in the architecture of the neural network meant to accept the input data and pass it onto the rest of the network.
- Hidden Layer - These can be multiple in number and are present between the input and output layers. They help improve performance and effectiveness, meanwhile, adding to the complexity. They perform various functions within the neural network such as the transformation of data into the relevant format, creating the necessary features from the dataset, etc.
- Output Layer - Final layer in the neural network holds the result, which is essentially the final output for the problem that is being solved.

C. Convolutional Neural Network (CNN)

Convolutional Neural Networks (CNNs) are a type of neural network which is extensively used in image classification [20]. The setup is made up of cells that behave as filters, which are able to extract features and support image classification. In malware analysis as well - CNNs are the preferred model for image classifiers. The basic architecture is composed of multiple layers-

- Convolutional Layer - functions as a feature extractor, applies convolutional filters and carries out feature matching.
- Activation Layer - uses an activation function such as a logistic function or softmax function to transform input from the first layer into a new output vector.
- Pooling layer - lies in the middle of 2 convolutional layers, helps improve efficiency and minimize the count of parameters, etc.
- Fully Connected Layer - final layer of any neural network, it receives the input vector and produces the final output vector

The function of analysis of the feature extraction is performed by several layers present in CNN architecture encoding. Feature extraction can be performed on any motif. Motifs are distinctive patterns that are subsets of the input image on which feature extraction is to be performed. Motifs additionally comprise sub-motifs that are repetitive and shared substructures in nature. Classification decisions are supported by the repetitiveness of convolutions. The step is performed by modifying the input image and filters at each layer and by modifying the feature map at the topmost layer.

D. K-Nearest Neighbour algorithm

K-Nearest Neighbour is a machine learning technique that is a supervised algorithm that classifies any new data point or entry as per its similarity with the existing store of available data. It does make any assumptions beforehand about the input data and is non-parametric in nature. It is extensively used in the training of models because it is able to classify new data into suitable categories as per the similarity of the new data with the stored data. This makes it a very popularly used model in classification and regression-based problems. Any new data point is compared with existing sets of data to carry out the best possible classification. This makes KNN very useful in malware classification as well when the converted malware files are passed as images into the model.

E. Support Vector Machines and their implementation when combined with CNN

Various research has shown that combining the successes of CNN and SVM to create a multi-class classification model has proved to be very beneficial. In the initial step, SVMs can handle image classification by analyzing the pixels of source images and using relevant margins. Then, CNN improves upon the classification done by SVM by using various blocks to carry out segmentation. An SVM layer is introduced instead of the softmax layer for classification. The SVM layer applies a linear classifier to the features extracted by the preceding layers. The combination helps improve the accuracy over traditional CNN models and provides greater nuances into the classification of malware families.

IV. PROPOSED METHODOLOGY

This passage describes a study that used convolutional neural networks (CNNs) to generate images of malware files for the purpose of classifying them into different families. The researchers used two datasets: the MalImg dataset, which con- tains 9,339 samples of malware from 25 different families, and the Microsoft Malware Dataset, which contains 21,741 files from 9 different malware classes. The datasets were divided into training and testing sets in a 70:30 ratio. The researchers also converted the malware files into byte files (hexadecimal representations) of the malware's portable executable (PE) portion using an IDE disassembler tool. These byte files were then converted into grayscale images, which were used as input for the CNN. The goal of the study was to classify the malware into different families based on the patterns present in the grayscale images following steps as below-

A. File Conversion

In the implementation of the model, the first step was to convert the malware's portable executable files into byte files using an IDE disassembler tool. The byte files represent the hexadecimal representation of the malware's PE portion. The dataset is divided into two sections: the instruction's memory address offset and the instruction itself. This division helps in the conversion of the executable files into byte files. The byte files are essential as they act as mappings of the original assembly instructions found in the malware's assembly source code files.

The malware binaries used in the dataset are in the portable executable format, which are typically recognized by file extensions such as .bin, .dll, and .exe. Portable executable files consist of various components, including the code section (.text) that contains program instructions, the read-only data section (.rdata), the modifiable data section (.data), and the resources section (.rsrc) used by the malware.To convert the binary files into grayscale images, hexadecimal pairs are combined to create pixel values. The conversion is done by taking 8 bits (1 byte) at a time. These grayscale images represent textual patterns that form the basis for classifying malware into different families.

Overall, the process involves disassembling the malware binaries, extracting the assembly instructions, converting them into byte files, and further transforming them into grayscale images for classification purposes.

The process can be summarized as follows:

1) Read binary malware file in the form of vector containing 8-bits unsigned integers.
2) Convert every component of vector from binary to decimal value.
3) Save the decimal value using another vector representing samples of malware file.
4) Reshape resultant vector to 2D/3D matrix and visualize as grayscale/ RGB image.
5) Determine spatial resolution of the binary file as per its size.

B. Model Architecture

Multiple layers, including Conv2D, MaxPooling2D, Flatten, and Dense layers, make up the model architecture. The Max-Pooling2D layers lower spatial dimensions while the Conv2D layers use filters to extract features from the input images. The Dense layers carry out high-level feature extraction and classification while the Flatten layer transforms the tensor into a 1D representation. Fig. 1 demonstrates chosen model architecture for classification of images.

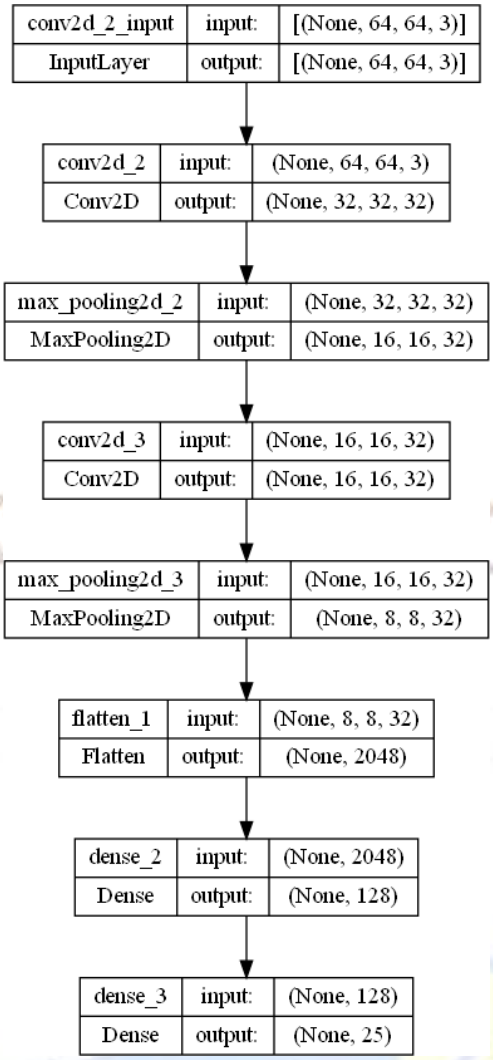

Fig. 1. CNN Architecture for Image Classification

C. Datasets

The following datasets are tested on a set of 3 algorithms - KNN, CNN and CNN + SVM -

1) *Malimg Dataset*: It contains 9,339 samples of malware provided by Nataranjan et al [15]. These samples are visual- ized as images before applying CNN on them. In our imple- mentation, each malware sample for testing and validation is a part of one of the malware families of the dataset. There are a total of 25 families. The families in the Malimg Dataset are pre-distributed into 2 parts: training and testing samples. Training samples are 70 percent of the dataset which translates to 6,538 malware samples, and testing data is 30 percent of the dataset which translates to 2,801 samples.

2) *Microsoft BIG Dataset:* As part of its malware classification challenge on Kaggle, Microsoft published a sizable collection of malware samples in 2015 that included 21,741 files. 10,686 malware files were split into training and testing portions in a 70:30 ratio in order to make implementation easier. 7,480 malware files were used for training and 3,206 for testing. Similar to Malimg, other datasets have distinct splits and distributions of malware samples. All 10,686 files, however, fall under one of 9 malware classes.

V. EXPERIMENTS AND RESULTS

The final comparison between the three different techniqueson the two datasets can be described as follows:

On the Malimg dataset, the CNN model achieved an accuracy of 88.11 percent, the KNN model with k=21 achieved 88.65 percent, and the CNN + SVM model achieved the highest accuracy of 96.57 percent. This indicates that the combination of CNN and SVM techniques outperforms this dataset's standalone CNN and KNN models. On the Microsoft (BIG) dataset, the CNN model achieved an accuracy of 86.46 percent, while the KNN model with k=3 had a lower accuracy of 77.21 percent. However, the CNN + SVM model achieved an accuracy of 94.80 percent, again showing that the combination of CNN and SVM techniques outperforms the standalone CNN and KNN models on this dataset.

Overall, the results of this study demonstrate the effective- ness of using a combination of CNN and SVM techniques for malware classification. The results suggest that this approach can provide a more robust and accurate solution compared to using CNN or KNN models alone. It is important to note that preliminary, non-exhaustive tests were carried out utilizing Genetic Algorithm (GA) approaches to optimize the classification models in addition to the above- mentioned findings. The accuracy in these tests reached up to 98 percent on the Malimg dataset, which is a considerable improvement over the present outcomes. These results, however, require thorough and substantiated testing and have been left out of the study's primary analysis. The performance of the malware

classification models may be improved further by the use of Genetic Algorithm (GA) approaches, which could lead to even more precise and reliable results. Future research is expected to explore this promising avenue in more depth, as well as focus on fine-tuning the parameters and exploring different architectures to optimize the performance of this approach.

VI. CONCLUSION

The research presented in this text focuses on developing techniques to classify and characterize malware using visual representations. Malware code files are converted to grayscale images and models for image classification like CNN, KNN, and CNN+SVM are used for implementation purposes. Results were obtained by comparing the effectiveness of these methodologies on the Microsoft BIG Malware Dataset and Malimg Dataset. Future research directions involve developing an evolutionary algorithm, specifically genetic programming, for malware classification as it has the potential to produce highly effective results and address drawbacks of traditional classification techniques.

TABLE I
DESCRIPTION OF MICROSOFT 2015 BIG DATASET

Family Type	Family Name	No. of Variants	Click Fraud	Send Spam SMS /Email	Malware Propogation	Credential Theft	Browser Hijacking
Worm	Ramnit	1541	✓	X	✓	✓	X
Adware	Lollipop	2478	✓	X	X	X	X
Backdoor	Kelihos_ver3	2942	X	X	✓	✓	X
Trojan	Vundo	475	✓	X	✓	✓	✓
Backdoor	Simda	42	✓	✓	✓ X ✓		
Trojan Downloader	Tracur	751	✓	X	✓	✓	X
Backdoor	Kelihos_ver1	398	X	X	✓	✓	X
Obfuscated Mal-ware	Obfuscator.ACY	1228	✓	X	✓	X	X
Backdoor	Gatak	1013	✓	✓	✓	X	X

TABLE II
DESCRIPTION OF MALIMG DATASET

Family Type	Family Name	No. of Variants	Click Fraud	Send Spam SMS	Malware Propogation	Credential Theft	Browser Hijacking
Dialer	Adialer.C	122	X	X	X	X	X
Backdoor	Agent.FYI	166	X	X	X	✓	X
Worm	Allaple.A	2949	X	X	✓	X	X
Worm	Allaple.L	1591	X	X	✓	X	X
Trojan	Alueron.gen!J	198	X	✓	✓	✓	✓
Worm:AutoIT	Autorun.K	106	X	X	✓	X	X
Trojan	C2Lop.P	146	X	X	X	✓	X
Trojan	C2Lop.gen!G	200	X	X	✓	✓	✓
Dialer	Dialplatform.B	177	X	X	X	X	X
Trojan Downloader	Dontovo.A	162	X	X	✓	✓	X
Rogue	Fakerean	381	X	X	X	X	✓
Dialer	Instantaccess	431	X	✓	X	X	X
PWS	Lolyda.AA 1	213	X	✓	✓	✓	X
PWS	Lolyda.AA 2	184	X	✓	✓	✓	X
PWS	Lolyda.AA 3	123	X	✓	✓	✓	X
PWS	Lolyda.AT	159	X	✓	✓	✓	X
Trojan	Malex.gen!J	136	X	✓	✓	✓	✓
Trojan Downloader	Obfuscator.AD	142	X	X	✓	X	X
Backdoor	Rbot!gen	158	X	X	✓	✓	X
Trojan	Skintrim.N	80	X	✓	✓	✓	X
Trojan Downloader	Swizzor.gen!E	128	X	X	✓	✓	X
Trojan Downloader	Swizzor.gen!I	132	X	X	✓	X	X
Worm	VB.AT	408	X	X	✓	X	X
Trojan Downloader	Wintrim.BX	97	X	X	✓	X	X
Worm	Yuner.A	800	X	X	✓	X	X

TABLE III
PERFORMANCE COMPARISON ON MALIMG AND MICROSOFT (BIG) DATASETS

Dataset	**CNN**	**KNN (k=21)**	**CNN + SVM**
Malimg	88.11%	88.65%	**96.57%**
Microsoft (BIG)	86.46%	77.21%	**94.80%**

TABLE IV
ACCURACY OF DIFFERENT APPROACHES

Approach	**Year**	**Accuracy (%)** (Malimg)	**Accuracy (%)** (BIG)
SVM-CNN	2023	96.57	94.80
Alexnet [16]	2021	90.5	85.82
Resnet-50 Network [16]	2021	97.5	89.56
Deep Neural Network [16]	2021	97.78	94.88
Dense CNN [17]	2019	94.64	93
CNN [18]	2018	94.5	93.4
CNN [19]	2017	93.57	93.92

VII. REFERENCES –

[1] N. Idika and A. P. Mathur, "A survey of malware detection techniques," Purdue University, vol. 48, 2007.

[2] K. Rieck, T. Holz, C. Willems, P. Dussel, and P. Laskov, "Learning and classification of malware behavior," in International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment. Springer, 2008, pp. 108–125.

[3] M. Siddiqui, M. C. Wang, and J. Lee, "A survey of data mining techniques for malware detection using file features," in Proceedingsof the 46th Annual Southeast Regional Conference on XX. ACM, 2008, pp. 509–510

[4] W. Hardy, L. Chen, S. Hou, Y. Ye, and X. Li, "Dlmd: A deep learning framework for intelligent malware detection," in Proceedings of theInternational Conference on Data Mining (DMIN), 2016

[5] M. G. Schultz, E. Eskin, F. Zadok, and S. J. Stolfo, "Data mining meth- ods for detection of new malicious executables," in IEEE Symposiumon Security and Privacy. IEEE, 2001, pp. 38–49.

[6] J. Z. Kolter and M. A. Maloof, "Learning to detect malicious executables in the wild," in Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2004, pp. 470–478.

[7] D. Gibert Llaurado, "Convolutional neural networks for malware clas- sification," Master's thesis, Universitat Politecnica de Catalunya, 2016.

[8] "Microsoft malware classification challenge (big 2015) first place team: Say no to overfitting," http://blog.kaggle.com/2015/05/26/microsoft- malware-winners-interview-1st-place-no-to-overfitting/

[9] G. Conti and S. Bratus, "Voyage of the Reverser: A Visual Study of Binary Species," presented at the Black Hat USA, 2010.

[10] R. Pascanu, J. W. Stokes, H. Sanossian, M. Marinescu, and A. Thomas, "Malware classification with recurrent networks," ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc., vol. 2015-Augus, pp. 1916–1920, 2015

[11] G. E. Dahl, J. W. Stokes, L. Deng, and D. Yu, "Large-scale malware classification using random projections and neural networks," ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc., pp. 3422–3426, 2013

[12] J. Kinable and O. Kostakis, "Malware classification based on call graph clustering," J. Comput. Virol., vol. 7

[13] M. Z. Shafiq, S. M. Tabish, F. Mirza, and M. Farooq, "PE-Miner : Mining Structural Information to Detect Malicious Executables inRealtime Agenda Introduc1on to Domain Problem Definition Proposed Solution," 2009.

[14] M. Kalash, M. Rochan, N. Mohammed, N. D. B. Bruce, Y. Wang, and F. Iqbal, "Malware Classification with Deep Convolutional Neural Net- works," in 2018 9th IFIP International Conference on New Technologies, Mobility and Security, NTMS 2018 - Proceedings, 2018, vol. 2018- Janua, pp. 1–5

[15] L. Nataraj, S. Karthikeyan, G. Jacob, and B.S. Manjunath, "Malware images: visualization and automatic classification," in Proceedings of the 8th International Symposium on Visualization for Cyber Security, VizSec '11, pp. 4:1–4:7, New York, NY, USA, 2011. ACM.

[16] Ö . Aslan and A. A. Yilmaz, "A New Malware Classification Framework Based on Deep Learning Algorithms," in IEEE Access, vol. 9, pp.87936-87951, 2021, doi: 10.1109/ACCESS.2021.3089586.

[17] A. Singh, A. Handa, N. Kumar, and S.K. Shukla, "Malware classification using image representation," in Cyber Security Cryptography and Ma- chine Learning: Third International Symposium, CSCML 2019, Beer- Sheva, Israel, June 27–28, 2019, Proceedings 3, pp. 75-92, Springer International Publishing, 2019.

[18] Z. Cui, F. Xue, X. Cai, Y. Cao, G.-g. Wang, and J. Chen, "Detection of malicious code variants based on deep learning," IEEE Transactions on Industrial Informatics, vol. 14, no. 7, pp. 3187-3196, 2018.

[19] J.-S. Luo and D. C.-T. Lo, "Binary malware image classification using machine learning with local binary pattern," in 2017 IEEE International Conference on Big Data (Big Data), pp. 4664-4667, IEEE, 2017.

[20] A. Dua, A. Bhatia, B. Kalra, and S. Vashishtha, ”A Novel Recurrent and Convolutional Neural Network Technique for Generating Handwriting from Voice,” in 2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA), pp. 1439-1444, IEEE, 2021.

[21] S. Vashishtha and S. Susan, ”Neuro-fuzzy network incorporating multi- ple lexicons for social sentiment analysis,” Soft Computing, vol. 26, no. 9, pp. 4487-4507, 2022.

[22] V. Gupta, H. Gaur, S. Vashishtha, U. Das, V.K. Singh, and D.J. Hemanth, ”A fuzzy rule-based system with decision tree for breastcancer detection,” IET Image Processing, vol. 17, no. 7, pp. 2083-2096, 2023.

[23] S. Vashishtha and S. Susan, ”Inferring sentiments from supervised classification of text and speech cues using fuzzy rules,” Procedia Computer Science, vol. 167, pp. 1370-1379, 2020.

[24] I. Jain and N. Goel, ”Advancements in Image Splicing and Copy-move Forgery Detection Techniques: A Survey,” in 2021 11th International Conference on Cloud Computing, Data Science Engineering (Conflu- ence), IEEE, 2021.

[25] S. Vashishtha, S. Kumar, V. Bothra, V. Singhal, and A. Sharma, ”Vehicle Detection System using YOLOv4,” in 2022 4th International Conferenceon Artificial Intelligence and Speech Technology (AIST), pp. 1-5, IEEE,December 2022.