Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 4, Number 1, May 2024
https://doi.org/10.47709/brilliance.v4i1.3869

Detection of Malware Threats in Internet of Things Using Deep Learning

Naufal Nashrullah [1*], Ari Purno Wahyu W[2]
[1,2]Informatics Department, Widyatama University, Indonesia
[1]naufal.nashrullah@widyatama.ac.id, [2]ari.purno@widyatama.ac.id

open access

*Corresponding Author

Article History:
Submitted: 16-05-2024
Accepted: 28-05-2024
Published: 10-06-2024

KEYWORDS:
CNN; Deep Learning; Internet of Things; Malware.

ABSTRACT

This paper examines the potential risks associated with the Internet of Things (IoT) as a new gateway for cyberattacks. The continuous access it provides to systems, applications, and services within organizations increases the likelihood of serious threats, such as software piracy and malware attacks, which can result in the theft of sensitive information and significant economic losses. To address these concerns, researchers have proposed the use of Deep Convolutional Neural Network (DCNN) to detect malware infections in IoT networks by using software plagiarism methods to identify source code similarities. Pre-processing techniques are used to break the source code into smaller pieces for in-depth analysis. Deep learning methods aim to find similar source code in different types of programming languages by using the TensorFlow framework. The malware samples were obtained from the Android Malware dataset on Kaggle. The proposed deep learning method, namely the Deep Convolutional Neural Network, was employed to detect malware infections in IoT networks. The experimental results showed that the combined approach obtained good classification results in overfitting and accuracy compared to the previous approach state. The tokenization process extracts keywords from the source code but does not display the internal appearance of the source code. Abstract syntax tree and control flow graph features to capture syntax flow and source code control.

INTRODUCTION

The Internet of Things (IoT) refers to the interconnection of physical objects through the internet, which are embedded with electronic chips, sensors, and other forms of hardware. Each unique device is globally identified by a radio frequency identifier (RFID). These smart objects communicate with other connected nodes and can be monitored and controlled remotely (Yadav et al., 2019). IoT devices can also be used for cyberattacks because they are always open and available on the network. The IoT-cloud industry is susceptible to malware and pirated software, which can be used for malicious purposes and compromise security (Karbab et al., 2017).

Malware is a computer program designed to cause damage to a system. Software piracy is the development of software by illegally using source code from other people's work and posing as the original version. Hackers can copy the logic of the original software by reverse engineering procedures and can design the same logic in other types of source code (IEEE Staff, 2018).

In 2022, the National Internet Emergency Response Center (CNCERT) compiled data indicating that the number of malicious programs exceeds 1,550,000 users and the number of malicious programs reaches 72,119,000. This has an impact on the security of the IoT-cloud industry. Furthermore, a 2016 report by the Business Software Alliance (BSA) indicated that the prevalence of public software piracy was approximately 39%. This resulted in annual business losses estimated at 52.2 billion dollars. Additionally, numerous studies have demonstrated that every piece of software contains source code that is emulated in a logical context to the extent of 5% (Adrienne et al., 2018).

The impact of malware infections on the security of the Internet of Things (IoT) cloud industry is significant. The presence of more than 1.55 million malicious programs infecting more than 72 million devices indicates a serious threat level to the IoT ecosystem. The IoT cloud industry is vulnerable to malware attacks because its devices are often directly connected to the internet and have wide access (Rafique et al., n.d.). Malware attacks can result in operational disruptions, the theft of sensitive data, or even physical damage to the IoT infrastructure, thereby threatening the overall security of the system and causing significant financial impacts for companies and consumers (Ullah et al., 2019). To address the aforementioned issues, deep learning will be proposed as a means of identifying pirated software attacks and malware-infected files within the IoT-cloud industry. In this paper, researchers propose the use of a deep convolutional neural network (DCNN) to detect malware infections in IoT networks by analyzing color image visualization. The malware samples are obtained from the Android Malware dataset from Kaggle. The proposed deep learning method utilizes a deep convolutional neural network (DCNN) to detect malware infections in IoT networks.

Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 4, **Number** 1, May 2024
https://doi.org/10.47709/brilliance.v4i1.3869

LITERATURE REVIEW

A multitude of studies have concentrated on the visualization of malware features with the objective of enhancing the efficacy of classification outcomes, as well as reducing the expenditure of time, size, and resources. For instance, convolutional neural networks (CNNs) and image-based malware classification techniques were introduced in (Hu et al., 2020; Koay et al., 2022). This method attained a classification accuracy of 98.52% and randomly selected 10% of samples for testing from malware families. The proposed deep learning model (Kumar et al., 2018) for malware detection achieved 98% classification accuracy for 9,339 malware samples. A convolutional neural network (CNN) model was proposed for malware classification (Wang et al., 2022) (Ji et al., 2019).

This paper will then propose the Deep Convolutional Neural Network (DCNN) method, which is a type of artificial neural network architecture that is highly effective in processing images and other spatial data. In the context of malware classification, the DCNN can be used to identify and classify types of malware based on features extracted from binary or opcode representations of executable files (Saxe & Berlin, 2015).

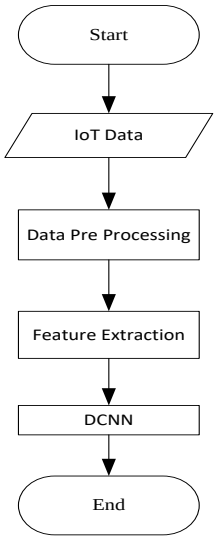

Figure 1. flowchart illustrating the malware security threat prediction architecture model.

METHOD

The deep convolutional neural network (DCNN) concept for malware classification involves the use of a deep and complex convolutional neural network (CNN) architecture to extract relevant features from binary data or opcodes of executable files. This paper presents a deep DCNN concept for malware classification (Dilhara, 2021).

Pre Process and Feature Extraction

The detection of pirated software is a challenging task due to the techniques employed by pirates to hide or modify the code, making it difficult to detect. To overcome this problem, we use software plagiarism methods to identify source code similarities. Pre-processing techniques are used to break the source code into smaller pieces for in-depth analysis. In the context of pirated software detection, techniques such as stemming, root word extraction, frequency extraction, and stop word removal can be highly beneficial in the analysis of source code text or other text related to such software. The application of these techniques is typically conducted in the text pre-processing stage prior to further analysis, such as classification or classification of pirated software detection. The utilization of these techniques can assist in enhancing the accuracy and efficiency of the pirated software detection process by reducing noise or irrelevant information in the analysis. In the breach phase, frequency (log TF) is employed (Association for Computing Machinery. Special Interest Group on Information Retrieval., 2013) (Haddi et al., 2013). Mathematically, TFIDF can be defined as in equation 1.

$$tfidf\ (t,d,D)\ =\ tf\ (t,d)\ x\ idf\ (t,D) \tag{1}$$

In this context, the symbol t denotes the sign, the symbol f denotes the number of frequencies, the symbol d denotes each individual document, and the symbol D denotes all documents used in the data set.

Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 4, Number 1, May 2024
https://doi.org/10.47709/brilliance.v4i1.3869

Deep Learning With Tensorflow Framework

TensorFlow is a machine learning tool utilized in complex computing. By employing specific TensorFlow APIs, one can utilize various types of machine and deep learning algorithms. It comprises a multitude of layers that can be customized to perform complex computations, train data, and track the path of each function (Abadi et al., 2016), (Baylor et al., 2017). The deep learning method is designed to identify similar source codes across different programming languages by leveraging the TensorFlow framework. Furthermore, the extracted similar codes are employed to identify instances of pirated software. The deep learning model utilizes these weight values. The dense layers, also referred to as fully connected layers, are configured to receive and output data. There are three dense layers with 100, 50, and 30 neurons, respectively. The first layer receives data using input variables with input shape parameters, and each neuron receives information from the previous layer, thus forming a tightly connected network. The fourth dense layer serves as the output variable and targets the plagiarized code.

The utilization of the dropout layer facilitates the enhancement of the deep learning approach with respect to the activation and loss functions, the optimizer, and the learning error rate. Additionally, the dropout layer serves to address the issue of overfitting. To generate data patterns that are deemed acceptable, the rectifier activation method (ReLU) is employed (Elfwing et al., 2017). Mathematically, this is demonstrated as the positive part of the argument, as illustrated in equation 2.

$$f\,(x)\ =\ x\ +=\ max(0,x) \tag{2}$$

In this context, X represents the input to the same neuron. The sigmoid function represents a successful logistic strategy for handling multi-class problems (Adrienne et al., 2018). It is mathematically described as follows:

$$s(x) = \frac{1}{1 + e^{-x}} \tag{3}$$

The sigmoid function, represented by S, is employed in this context. Adam's optimizer, also known as stochastic gradient descent, is utilized to optimize deep learning models. It determines the adaptive learning speed in accordance with all obstacles (Adrienne et al., 2018), (Kumar et al., 2018). Equations 4 and 5 these equations illustrate the average decay of the quadratic fitting gradient.

$$mt = \beta 1 mt{-}1 + (1 - \beta 1) gt \tag{4}$$

$$vt = \beta 2 vt{-}1 + (1 - \beta 2) g^2 \tag{5}$$

The formula for calculating the average of the two instant gradients, denoted by Mt and Vt, respectively, is as follows:

Mt = (Mt-1 + Mt)/2
Vt = (Vt-1 + Vt)/2

This formula is employed by Adam's optimization algorithm to compute the average of the predecessor and successor gradients. The running exponential average of the first gradient, Mt, and the square gradient, Vt, is then updated using these estimates.

One of the contributions of Tensorflow-based algorithms is as follows:

The design and extension of machine learning approaches for large-scale data incorporating diverse types of computing APIs, such as GitHub frameworks. The system is capable of automatically training models based on input, hidden, and output layers with various activation functions. These algorithms can be configured and run on a range of devices, from small to large, and offer reliable services while updating and extending the designed models.

RESULT

The accuracy obtained from this method reaches 98%, although it is still below that of the Sequential Neural Network. However, the Deep Convolution Neural Network method can overcome the problem of overfitting, as can be demonstrated in Figure 2. For the Sequential Neural Network, the blue and orange curves are inversely proportional, beginning with period 3.

Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 4, **Number** 1, May 2024
https://doi.org/10.47709/brilliance.v4i1.3869

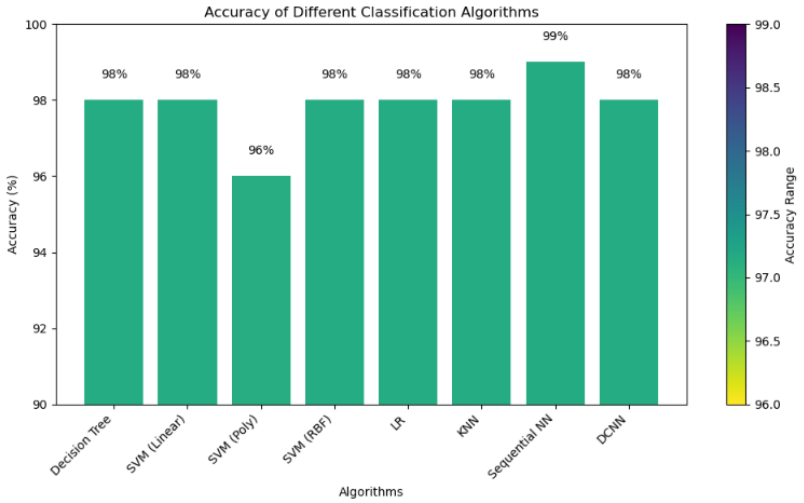

Figure 2. A comparison of malware detection approaches with other methods based on classification accuracy

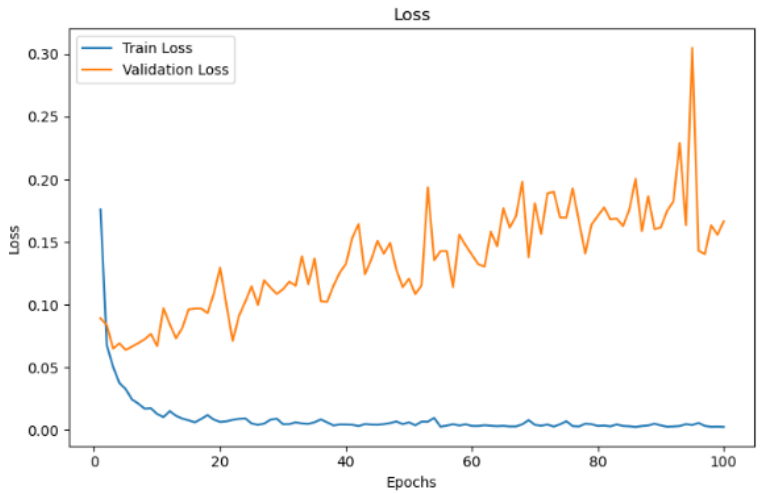

Figure 3. Plotting dynamic loss Sequential Neural Network

DISCUSSION

One can employ software plagiarism measures to identify code similarities in pirated software. To analyze the suggested methods for software piracy, we retrieved source codes from the Android Malware dataset from Kaggle. First, we collected useful tokens for each source with frequency details through the data set processing process. This process included stemming, root word, maximum and minimum token length, maximum and minimum token frequency, and so forth. Secondly, to extract token weighting, feature selection and extraction methods such as Term Frequency and Inverse Document Frequency (TFIDF) and Logarithm Term Frequency (LogTF) are employed. The weighting method increases the contribution of each token in the document or the entire document.

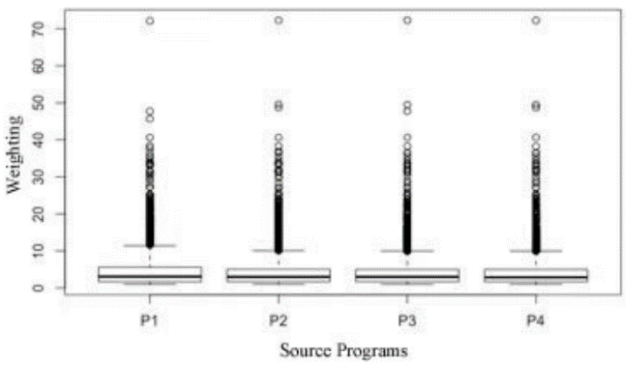

Figure 4. Plot of the weight values. (Ullah et al., 2021)

Brilliance
Research of Artificial Intelligence
E-ISSN : 2807-9035
Volume 4, Number 1, May 2024
https://doi.org/10.47709/brilliance.v4i1.3869

Figure 4 presents a box plot of the weight values, with P1, P2, P3, and P4 on the x-axis representing the programming solutions of the four questions, respectively, and the y-axis displaying the weight values for each source code. Each program addresses four distinct programming problems, resulting in four input variables in the first dense layer. The second and third layers are configured as hidden layers with the objective of improving accuracy. The dropout layer is configured with the input and each hidden layer in order to address the issue of overfitting.

Furthermore, the number of neurons in different layers with different activation functions and learning error rates can be fine-tuned to improve classification accuracy. Figure 2 illustrates the dynamic visualization of accuracy, validation accuracy, loss, and validation loss in percentage size. The blue curve represents the loss, while the green curve represents the validation loss. Initially, the blue curve commences from the identical 0.6 point, while the orange curve commences from 0.3. Up to period 3, both curves exhibit a similar pattern, namely a decrease. However, following this, the blue curve experiences fluctuations relative to the orange curve. As both curves are decreasing, there is a reduced probability of an overfitting issue. Conversely, if these curves increase or behave in an opposing manner, then overfitting is likely to occur. A comparison is presented between the proposed approach and other learning techniques, as illustrated in Figure 5.

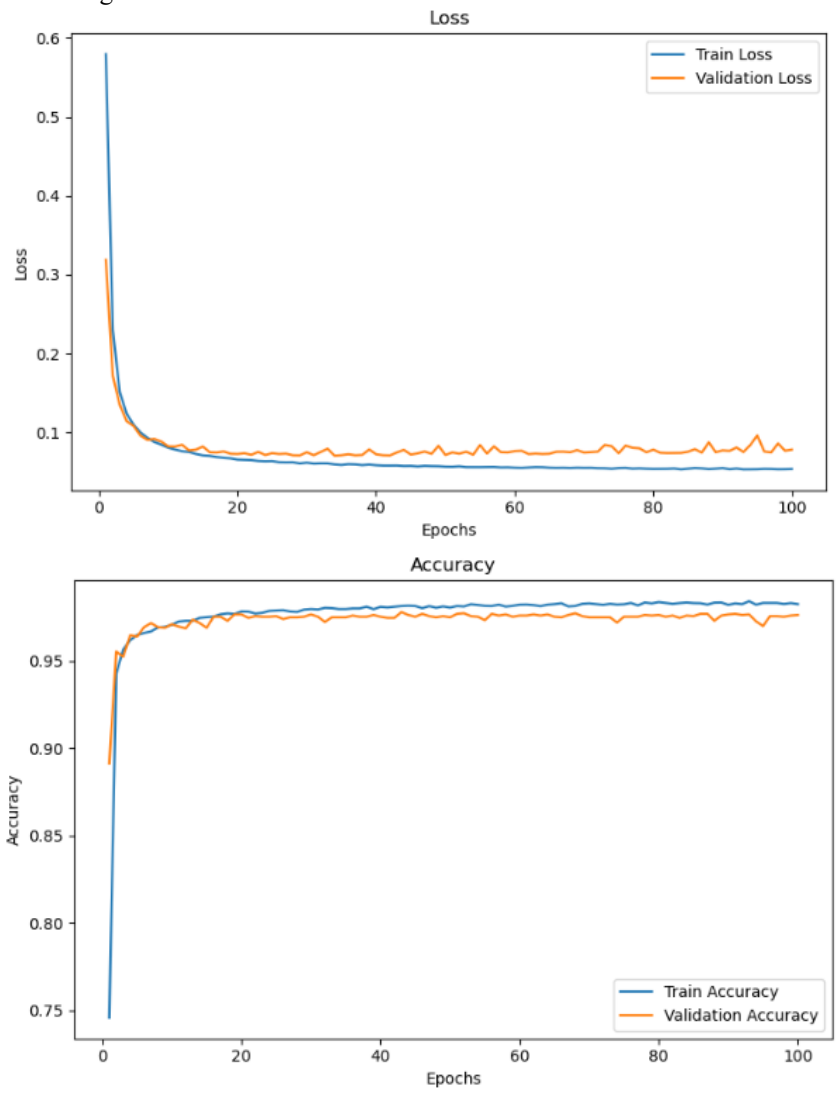

Figure 5. Plotting dynamic loss dan Plotting dynamic accuracy

CONCLUSION

The growth of IoT-based industrial networks is expected to be rapid in the future. One challenge in the field of cybersecurity using IoT-based big data is the detection of software piracy. We propose a deep learning-based combined approach to identify pirated files and malware. First, a TensorFlow neural network is proposed to detect pirated features from original software using software plagiarism. We use a dataset from Android Malware to investigate the proposed approach. The source code was pre-processed to remove noise and to further capture high-quality features that include

E-ISSN : 2807-9035
Volume 4, **Number** 1, May 2024
https://doi.org/10.47709/brilliance.v4i1.3869

useful tokens. Then, TFIDF and LogTF weighting techniques were employed to amplify the impact of each feature in terms of source code similarity. The weight values were then input to the designed deep learning approach.

The experimental results demonstrate that the combined approach yields satisfactory classification outcomes in terms of overfitting and accuracy, outperforming the previous state of the art. However, further evaluation is necessary. The tokenization process extracts keywords from the source code, but does not provide an internal view of the source code. The abstract syntax tree and control flow graph feature enables the capture of the syntax and control flow of the source code.

REFERENCES

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., … Zheng, X. (2016). *TensorFlow: A system for large-scale machine learning*. http://arxiv.org/abs/1605.08695

Adrienne, A., Tato, N., Nkambou, R., & Tato, A. (2018). *Workshop track-ICLR 2018 IMPROVING ADAM OPTIMIZER*. https://doi.org/10.13140/RG.2.2.21344.43528

Association for Computing Machinery. Special Interest Group on Information Retrieval. (2013). *SIGIR '13 : the proceedings of the 36th International ACM SIGIR Conference on Research & Development in Information Retrieval : July 28-August 1, 2013, Dublin, Ireland*. ACM.

Baylor, D., Breck, E., Cheng, H. T., Fiedel, N., Foo, C. Y., Haque, Z., Haykal, S., Ispir, M., Jain, V., Koc, L., Koo, C. Y., Lew, L., Mewald, C., Modi, A. N., Polyzotis, N., Ramesh, S., Roy, S., Whang, S. E., Wicke, M., … Zinkevich, M. (2017). TFX: A TensorFlow-based production-scale machine learning platform. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *Part F129685*, 1387–1395. https://doi.org/10.1145/3097983.3098021

Dilhara, B. A. S. (2021). Classification of Malware using Machine learning and Deep learning Techniques. *International Journal of Computer Applications*, *183*(32), 12–17. https://doi.org/10.5120/ijca2021921708

Elfwing, S., Uchibe, E., & Doya, K. (2017). *Sigmoid-Weighted Linear Units for Neural Network Function Approximation in Reinforcement Learning*. http://arxiv.org/abs/1702.03118

Haddi, E., Liu, X., & Shi, Y. (2013). The role of text pre-processing in sentiment analysis. *Procedia Computer Science*, *17*, 26–32. https://doi.org/10.1016/j.procs.2013.05.005

Hu, J., Shen, L., Albanie, S., Sun, G., & Wu, E. (2020). Squeeze-and-Excitation Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *42*(8), 2011–2023. https://doi.org/10.1109/TPAMI.2019.2913372

IEEE Staff. (2018). *2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*. IEEE.

Ji, W., Chen, G., Xu, B., Meng, X., & Zhao, D. (2019). Recognition Method of Green Pepper in Greenhouse Based on Least-Squares Support Vector Machine Optimized by the Improved Particle Swarm Optimization. *IEEE Access*, *7*, 119742–119754. https://doi.org/10.1109/ACCESS.2019.2937326

Karbab, E. B., Debbabi, M., Derhab, A., & Mouheb, D. (2017). *Android Malware Detection using Deep Learning on API Method Sequences*. http://arxiv.org/abs/1712.08996

Koay, K. Y., Tjiptono, F., & Sandhu, M. S. (2022). Predicting consumers' digital piracy behaviour: does past experience matter? *International Journal of Emerging Markets*, *17*(9), 2397–2419. https://doi.org/10.1108/IJOEM-09-2020-1067

Kumar, R., Xiaosong, Z., Khan, R. U., Ahad, I., & Kumar, J. (2018). Malicious code detection based on image processing using deep learning. *ACM International Conference Proceeding Series*, 81–85. https://doi.org/10.1145/3194452.3194459

Rafique, M. F., Ali, M., Qureshi, A. S., Khan, A., & Mirza, A. M. (n.d.). *Malware Classification using Deep Learning based Feature Extraction and Wrapper based Feature Selection Technique*. https://github.com/cyberhunters/Malware-Detection-Using-Machine-Learning.

Saxe, J., & Berlin, K. (2015). *Deep Neural Network Based Malware Detection Using Two Dimensional Binary Program Features*. http://arxiv.org/abs/1508.03096

Ullah, F., Naeem, H., Jabbar, S., Khalid, S., Latif, M. A., Al-Turjman, F., & Mostarda, L. (2019). Cyber security threats detection in internet of things using deep learning approach. *IEEE Access*, *7*, 124379–124389. https://doi.org/10.1109/ACCESS.2019.2937347

Ullah, F., Wang, J., Farhan, M., Habib, M., & Khalid, S. (2021). Software plagiarism detection in multiprogramming languages using machine learning approach. *Concurrency and Computation: Practice and Experience*, *33*(4). https://doi.org/10.1002/cpe.5000

Wang, S., Wang, J., Song, Y., Li, S., & Huang, W. (2022). Malware Variants Detection Model Based on MFF–HDBA. *Applied Sciences (Switzerland)*, *12*(19). https://doi.org/10.3390/app12199593

Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 4, **Number** 1, May 2024
https://doi.org/10.47709/brilliance.v4i1.3869

Yadav, E. S., Srinivasan, C. R., Saikalyan, P., & Premsagar, K. (2019). A review on the different types of internet of things (IoT). In *Article in Journal of Advanced Research in Dynamical and Control Systems* (Vol. 11, Issue 1). https://www.researchgate.net/publication/332153657