The Indonesian Journal of Computer Science
www.ijcs.net
Volume 13, Issue 5, October 2024
https://doi.org/10.33022/ijcs.v13i5.4349

Network Attack Classification using Neural Network-Based Imputation Technique

Safrizal Ardana Ardiyansa[1], Natasha Clarissa Maharani[2], Eric Julianto[3], Haidar Ahmad Fajri[4]
safrizal.braincore@gmail.com[1], natashaclarissa11@gmail.com[2], ericjuliantooo@gmail.com[3], fajrihaidar07@gmail.com[4]
[1,2,4]Department of Mathematics, Brawijaya University, Malang, Indonesia
[1,2,3,4]Brainore Indonesia, Jakarta, Indonesia

Article Information

Received : 30 Jul 2024
Revised : 7 Sep 2024
Accepted : 10 Oct 2024

Keywords

network attacks, machine learning, imputation technique, neural network.

Abstract

Rapid technological developments have changed access to information significantly, especially in telecommunications. This growth creates new threats, such as network attacks, so detection becomes critical for network security. Leveraging machine learning algorithms to detect threats is promising, with effectiveness largely dependent on selecting relevant features optimized by the bat algorithm. Data imputation is critical in preparing data sets, and neural network-based imputation techniques demonstrate outstanding performance, achieving accuracy rates of 99.4% on validation data and 99.3% on test data. This method consistently maintains precision, recall, and scores around 98%. Models using this method also approach perfection in classifying normal and Neptune labels. This imputation method can also be applied to other model architectures using autoML. Alternative models such as Light GBM, XGBoost, Random Forest, Extra Trees, and Weighted Ensemble L2 also exhibit exceptional accuracy, exceeding 99.8%.

A. Introduction

Technology has a significant impact on our daily lives [1]. It is growing and progressing at a rapid pace [2]. It also changes every little thing people do [3], such as access to information, learning, and communicating [4], [5]. Technology has also changed Indonesia [1]. Fast-paced technological development has paved the way to prepare for advanced technology in the future. This also includes the telecommunications sector [6]. Telecommunications is very important for Indonesia's development [7]. It has a significant impact on increasing connectivity [8], access to information, and community empowerment. The Indonesian government has been actively pursuing initiatives to advance the country's telecommunications sector, aiming to realize the "Indonesia Emas 2045" vision.

Unfortunately, as telecommunications grows, a new problem arises as well. This new problem is called network attacks [9]. In the context of computer networks, an attack refers to an effort to illicitly obtain, incapacitate, damage, modify, or gain unauthorized entry to or control over a resource [10]. There is a possibility that network attacks can slow down services or cause them to be unavailable for an extended period of time [11]. Users and network administrators must identify these attacks proactively to prevent any harm to the system [12]. Hence, it is imperative to offer effective approaches for recognizing, safeguarding against, and upholding network security to ensure its integrity.

Identifying network attack types detection is a suitable solution for this situation [13]. Early detection can help prevent potential damage, mitigate risks, and assist in finding the right solutions. By utilizing machine learning algorithms, a real-time automated and effective detection system can be achieved [14]. Machine learning algorithm has been successfully implemented in this topic [15]. The methods that are used are decision tree, ripple rule, random forest, and bayesian network [16]; multivariate correlation analysis [17]; naïve bayes classifier [18]; support vector machine and deep multi-layer perceptron [19]; AdaBoost [20]; and many others. This makes it a promising and efficient tool for network attack type identifiers.

The performance of machine learning relies on attribute selection [21], so it is essential to choose the most relevant features carefully [22]. Feature selection is useful when dealing with high-dimensional data [23]. Some reasons for performing feature selection are removing irrelevant features [24], increasing the predictive accuracy of learned models [25], reducing the computational cost of the data [22], and improving learning efficiency [25]. However, selecting the right features is a complex problem [26], and trying out all possible combinations is time-consuming [27]. The bat algorithm has shown remarkable advantages over other optimizations for select features. It takes inspiration from the echolocation behavior of micro bats [28]. It also utilizes intelligent search strategies that efficiently explore the search space [29].

The available dataset is not always ready for use. Missing values are a common issue in dataset problems, and that's why data imputation is necessary before building a machine learning model. Imputation is the process of filling in missing data with meaningful values. Imputing missing data is a mandatory step, as any data analysis can only be performed with complete datasets [30]. In categorical columns, imputing with the mode is one of the simplest and most straightforward methods

for estimating missing values [31]. For float or integer columns, imputation with the median can be used because it is robust against outliers [32]. However, at higher rates of missing data (30%, 40%, and 50%), the performance of neural networks for imputation is notably better than other methods [33]. For these reasons, this article proposes using a neural network-based imputation technique for network attack classification.

B. Research Method

The analytical process starts with dataset acquisition, followed by a comprehensive dataset exploration phase to understand its structure and characteristics. Subsequently, data preprocessing tasks, including cleaning and transformation, are undertaken to prepare the data for analysis. Missing data points are then imputed using Artificial Neural Networks (ANN), and feature selection is performed employing the bat algorithm to identify the most relevant variables. Finally, a machine learning model is developed to utilize the processed data for predicting network attacks. In summary, the research steps can be visualized in **Figure 1** below.

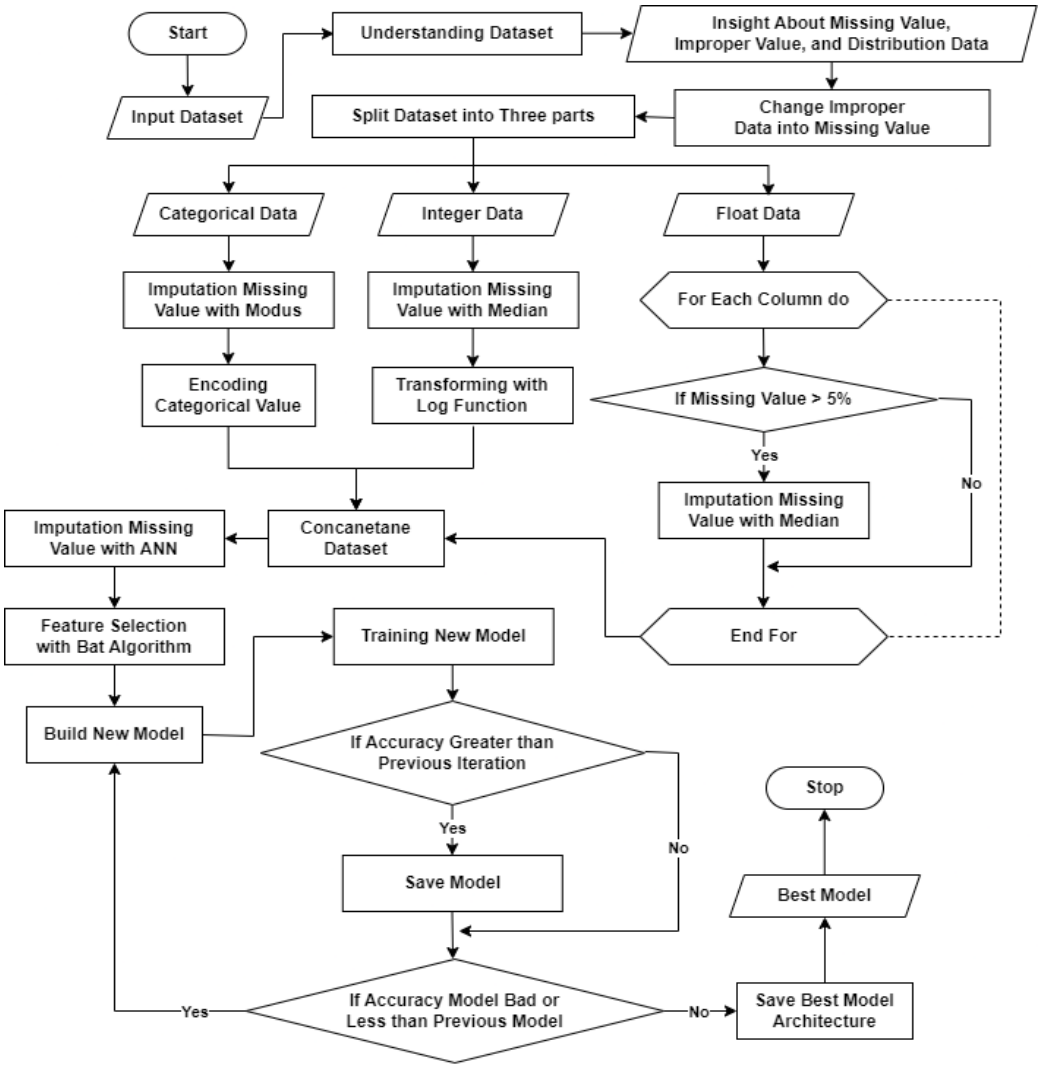

Figure 1. Research methodology

The initial step involves uploading the dataset to Google Colaboratory. The dataset adheres to a structured tabular format, which lends well to analysis and manipulation. Notably, this dataset is conveniently formatted in CSV (Comma-Separated Values) format, simplifying importing it directly into Google Colaboratory without encountering compatibility issues. The dataset encompasses a wealth of information on network traffic, encompassing a total of 112,446 rows and 41 columns. These columns encompass a diverse features, including duration, protocol type, service, flag, and source bytes, as shown in **Figure 2**. Additionally, one crucial column labeled as type of attack, serves as the target variable to predicted.

Sample of The Dataset

	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	...	dst_host_srv_count	type_of_attack
0	0	tcp	private	SH	0	0	0	0	0	0	...	1	nmap
1	0	tcp	private	S0	0	0	0	0	0	0	...	5	neptune
2	0	tcp	http	SF	285	3623	0	0	0	0	...	228	normal
3	0	tcp	http	SF	232	584	0	0	0	0	...	255	normal
4	1	tcp	smtp	SF	1080	327	0	0	0	0	...	154	normal

Figure 2. Sample of the dataset

This dataset has several problems, such as missing values, duplicate data, inappropriate values, and imprecise data distribution as seen in **Figure 3**. To overcome this, inappropriate values were converted into missing values and duplicate data were removed. Next, the dataset is divided into three parts, each containing columns with the same data type.

Figure 3. Skewed distribution of the count column from the dataset

The categorical dataset contains small missing values, so it will be imputated with mode. Because categorical data consists of non-numeric values, so encoding is applied to each column. The integer dataset also contain small missing value, so imputation process can be applied, but it contains a significant outlier as seen in **Figure 4**. Therefore, the median was chosen to fill in the missing value because it's less affected by outliers. Then the number outbound cmds column contains only one unique value, so it will be removed to prevent unnecessary redundancy. Most columns in integer datasets have right-skewed distributions due to the presence of outliers. To address this issue, logarithmic transformation has been applied to each integer-type column. The transformation is performed using the equation:

$$f(x) = \log(x + 1). \quad (1)$$

This logarithmic transformation serves to narrow the data distribution, effectively reducing the impact of outliers. Applying this transformation aims to make the data distribution less wide.

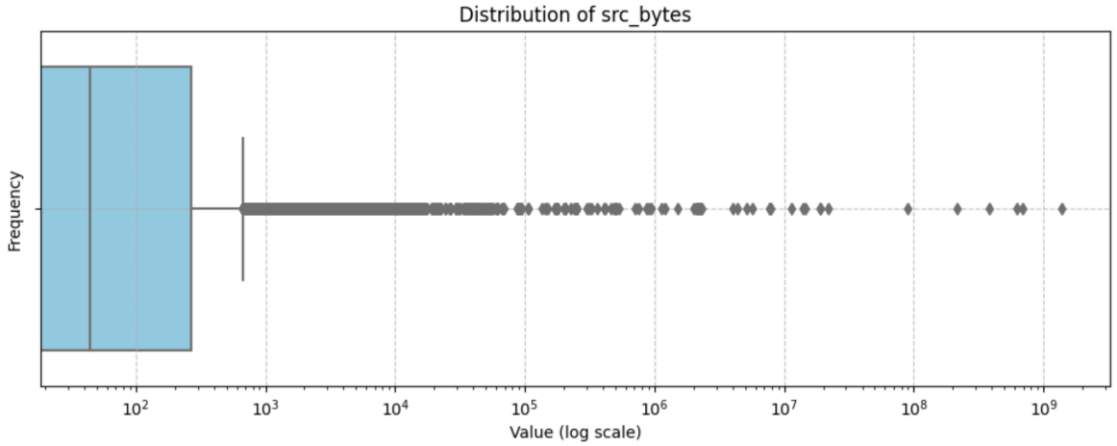

Figure 4. Distribution of the source bytes column

In float dataset, there are some columns that contain a notably high number of missing values as shown in **Table 1**. To ensure robust and reliable imputation, median imputation applied for columns that have small missing values and imputation neural network for the rest. The next step is merge all datasets with neural network. It can effectively do predictive tasks, involving impute the missing data. The neural network is denoted by the architecture as $a = (N, \varphi)$, where $L \in \mathbb{N}$ as the number of layers, $N \in \mathbb{N}^{L+1}$, and activation function $\varphi: \mathbb{R} \to \mathbb{R}$. It relies on the concept of neurons, which are fundamental units responsible to compute input by aggregating it and producing an output based on an activation function. The quantity of neurons in the input, output, and ℓ-th hidden layer is represented by N_0, N_L, and N_ℓ respectively. Let the number of parameters in neural network denote by $P(N) \coloneqq \sum_{\ell=1}^{L} N_\ell N_{\ell-1} + N_\ell$, then there is a corresponding function $\Phi : \mathbb{R}^{N_0} \times \mathbb{R}^{P(N)} \to \mathbb{R}^{N_L}$, which satisfies for every input $x \in \mathbb{R}^{N_0}$ and parameters:

$$\theta = \left(\theta^{(\ell)}\right)_{\ell=1}^{L} = ((\omega^{(\ell)}, b^{(\ell)}))_{\ell=1}^{L} \in \bigtimes_{\ell=1}^{L} (\mathbb{R}^{N_\ell \times N_{\ell-1}} \times \mathbb{R}^{N_\ell}) \cong \mathbb{R}^{P(N)} \qquad (2)$$

that $\Psi_a(x, \theta) = \Psi^{(L)}(x, \theta)$, where

$$\begin{aligned} \Psi^{(1)}(x, \theta) &= \omega^{(1)} x + b^{(1)}, \\ \bar{\Psi}^{(\ell)}(x, \theta) &= \varphi(\Psi^{(\ell)}(x, \theta)), \quad \ell \in \{1, 2, \dots, L-1\}, \text{and} \\ \Psi^{(\ell+1)}(x, \theta) &= \omega^{(\ell+1)} \bar{\Psi}^{(\ell)}(x, \theta) + b^{(\ell+1)}, \quad \ell \in \{1, 2, \dots, L-1\}, \end{aligned} \qquad (3)$$

and φ is applied componentwise. $\omega^{(\ell)} \in \mathbb{R}^{N_\ell \times N_{\ell-1}}$ and $b^{(\ell)} \in \mathbb{R}^{N_\ell}$ referred to weight matrices and bias vectors, $\bar{\Psi}^{(\ell)}$ and $\Psi^{(\ell)}$ represent the states of the N_ℓ neurons in the ℓ-the layer, both in terms of their activations and pre-activations. During the imputation process, priority is given to columns with the fewest missing values, that is dst host srv rerror rate column which has a missing value 4.92%, then dst host srv serror rate column with a missing value 5.24%, and so forth.

Table 1. Features in the float dataset with over 5000 missing values

Column Name	Missing Value	Precentage
same_srv_rate	42,110	37.92%
diff_srv_rate	41,920	37.75%
srv_diff_host_rate	18,316	16.50%
dst_host_srv_diff_host_rate	34,133	30.74%
dst_host_srv_serror_rate	5,821	5.24%
dst_host_rerror_rate	10,755	9.69%
dst_host_srv_rerror_rate	5,467	4.92%

The next step involves the feature selection process using the bat algorithm. In this algorithm, an iterative approach is employed to identify the feature subset that yields the highest accuracy. The positions of bats represent the selected features. The update of new solutions $\boldsymbol{x}_i^t$ and velocities $\boldsymbol{v}_i^t$ at time t is given in formula:

$$\begin{aligned} f_i &= f_{min} + (f_{max} - f_{min})\beta \\ \boldsymbol{v}_i^t &= \boldsymbol{v}_i^{t-1} + (\boldsymbol{x}_i^t - \boldsymbol{x}_*)f_i \\ \boldsymbol{x}_i^t &= \boldsymbol{x}_i^{t-1} + \boldsymbol{v}_i^t, \end{aligned} \tag{4}$$

with $\beta \in [0, 1]$ is a random number from a uniform distribution and $\boldsymbol{x}_*$ represent the position with the best fitness value among all bats. In this case, will used $f_{min} = 0$ and $f_{max} = 1$ since the solution domain is binary numbers. The exploitation procedure generates a new solution based on the neighborhood of the best solution, as defined by formula:

$$\boldsymbol{x}_{new} = \boldsymbol{x}_{old} + \frac{\varepsilon}{n}\sum_{i=1}^{n} A_i^t, \tag{5}$$

with $\varepsilon \in [-1,1]$ is a random number, n is total population, and A_i^t is the loudness of bat i-th at time t [34]. Loudness (A_i) and pulse rate (r_i) also updated as the iterations progress, as typically, loudness decreases after a bat finds prey, while the pulse rate increases. Initial loudness can be initialized with $A_0 = 1$ and $A_{min} = 0$, assuming that $A_{min} = 0$ means a bat has just found prey and temporarily stops emitting sound. The update of loudness and pulse rate is defined using

$$A_i^{t+1} = \alpha A_i^t, r_i^{t+1} = r_i^0[1 - \exp(-\gamma t)], \tag{6}$$

with α and γ being constants. For each $0 < \alpha < 1$ and $\gamma > 0$, can obtain by

$$A_i^t \to 0,\ r_i^t \to r_i^0,\ \text{when } t \to \infty. \tag{7}$$

In this case, the parameters that are used are $\alpha = \gamma = 0.9$. The initial loudness A_i^0 defined within the range $A_i^0 \in [1, 2]$, while the initial pulse rate r_i^0 within the range $r_i^0 \in [0, 1]$.

In this case, the search space is modeled in an n-dimensional space with boolean values $\{b_1, b_2, \dots, b_n\}$, where $b_i = 1$ or 0 for $i = 1, 2, \dots, n$. Because bat's position is represented by a binary vector, so a bat's position is constrained to be binary using formula:

$$S(v_i^j) = \left(1 + e^{v_i^j}\right)^{-1}, \tag{8}$$

where j represent j-th feature. The update of a bat's position is then revised with equation:

$$x_i^j = \begin{cases} 1, & S(v_i^j) > \sigma \\ 0, & S(v_i^j) \leq \sigma, \end{cases} \quad (9)$$

where $\sigma \sim U(0,1)$ [35]. The fitness of these binary bats is determined by their accuracy when employed in conjunction with a Random Forest model to identify the best feature subset.

The last step is building classifier model to predict the types of attacks occurring on network. This process is carried out iteratively, where the model is developed and evaluated multiple times until an architecture that yields the best validation accuracy is found. When the validation accuracy improves, the model is saved for further use. Using this approach ensures that the resulting model performs optimally in predicting network attack types.

C. Result and Discussion

The process starts with data input, followed by a preprocessing step to handle improper entries replaced with missing values. The data is then categorized into three types: categorical, integer, and float. Mode imputation is applied to fill missing values in categorical data and convert categories into numerical format. Median imputation is used for integer data with missing values, while for float data with more than 5% missing values, median imputation is employed. Finally, the three types of data are merged into a unified dataframe. With the transformation of categorical data into numerical format, every column now contains numerical data, and their value range is no longer excessively large, as shown in **Figure 5**.

Preprocessed data

	duration	src_bytes	dst_bytes	wrong_fragment	urgent	hot	...	is_host_login	is_guest_login	type_of_attack	dst_host_srv_rerror_rate
0	0.000000	0.000000	0.000000	0.0	0.0	0.0	...	0	0	3	0.0
1	0.000000	0.000000	0.000000	0.0	0.0	0.0	...	0	0	2	0.0
2	0.000000	5.655992	8.195334	0.0	0.0	0.0	...	0	0	4	0.0
3	0.000000	5.451038	6.371612	0.0	0.0	0.0	...	0	0	4	0.0
4	0.693147	6.985642	5.793014	0.0	0.0	0.0	...	0	0	4	0.0

Figure 5. Sample of the preprocessed data

Based on the binary bat and random forest algorithms used on the internet dataset, with the parameters set as follows: population size of 30, maximum iterations of 30, loudness for each bat at 0.9, and pulse rate for each bat at 0.9, the results show that the features that yield the highest accuracy on the validation data are duration, source bytes, wrong fragment, hot, number failed logins, number access files, count, srv count, srv serror rate, dst host srv count, dst host same srv rate, dst host serror rate, protocol type, service, flag, land, logged in, is host login, is guest login, dst host srv rerror rate, dst host srv serror rate, srv diff host rate, dst host srv diff host rate, diff srv rate, and same srv rate. These features result in a training accuracy of 93.66% and a validation accuracy of 93.38%.

The constructed model showcases remarkable training performance, achieving an accuracy of 99.4% and a low loss of 0.0167. This emphasizes its effective learning from the training data. Additionally, the model demonstrates strong generalization on the test set, with an accuracy of 99.3% and a validation loss of 0.03 as shown in **Figure 6**. This signifies its capability to make accurate

predictions on new, unseen data. Moreover, the accompanying graph illustrates consistent improvement in accuracy and loss with each epoch, highlighting substantial progress in the model's performance.

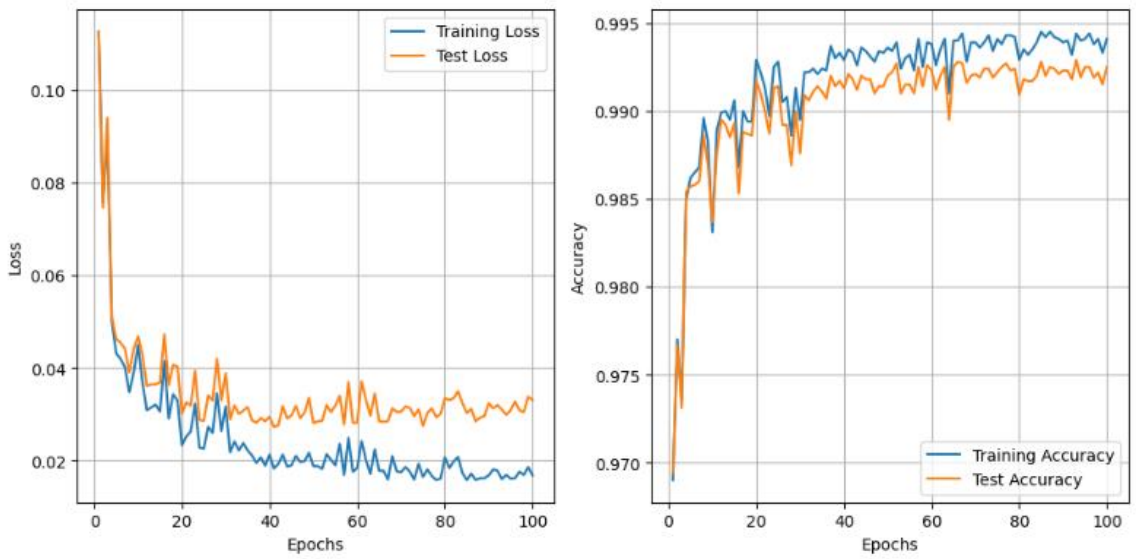

Figure 6. Graph of accuracy and loss on ANN model

Based on the evaluation of the model using the validation set, ANN obtained confusion matrix as shown in **Figure 7** with precision, recall, and F_1-score as seen in **Table 2**. Based on the confusion matrix, the ANN model can classify types of internet attacks quite well. ANN model can predict neptune and normal label perfectly, because it has perfect score on F_1-score. ANN also has average precision score at 0.98, average recall score at 0.97, and average F_1-score at 0.97. However, the ANN model is still not performing very well in predicting the nmap attack label, but it still quite well. It because ANN model has F_1-score at 0.88 on nmap label, which is smaller than the average of all labels.

		Prediction Label							
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Actual Label	**0**	345	0	1	0	0	0	1	0
	1	0	587	0	6	6	1	0	0
	2	0	0	7415	0	3	0	0	0
	3	0	33	0	220	6	0	0	0
	4	1	2	0	5	11936	0	3	4
	5	0	0	3	0	3	521	2	0
	6	0	0	0	0	14	2	580	0
	7	0	0	0	0	5	0	0	503

Figure 7. Confusion Matrix Proposed Model

Table 2. F_1-score, recall and precision ANN model

Label	Types of Attack	Precision	Recall	F_1- score
0	denial of service attack	0.99	0.99	0.99
1	ipswee	0.94	0.96	0.95
2	neptune	1.00	1.00	1.00
3	nmap	0.92	0.84	0.88
4	normal	1.00	1.00	1.00
5	portsweep	0.99	0.99	0.99
6	satan	0.98	0.98	0.98
7	smurf	1.00	0.99	0.99
	Macro average	0.98	0.97	0.97
	Weighted average	0.99	1.00	0.99

When comparing between other models with autoML, it was found that the accuracy of the train data for the neural network is higher than Cat Boost, Light GBMXT, and KNN. However, it is below that of Random Forest, Extra Trees, Light GBM, Weighted Ensemble L2, and XG Boost, as seen in **Table 3**. This indicates that the neural network is balanced when learning patterns. This makes the neural network's performance very high in predicting validation data. However, the computational time during the training process is longer than other models.

Table 3. Comparison between model architecture

Model Architecture	Train accuracy	Validation accuracy	Prediction time train	Prediction time val.	Training time
Random Forest Entropy	99.81%	99.80%	0.6662s	0.2437s	34.1958s
Extra Trees Entropy	99.78%	99.80%	0.7878s	0.1714s	19.5631s
Extra Trees Gini	99.78%	99.80%	0.7879s	0.1810s	21.4811s
Random Forest Gini	99.77%	99.84%	0.6885s	0.1615s	34.7532s
Light GBM Large	99.76%	99.88%	1.3905s	0.2204s	14.7312s
Light GBM	99.73%	99.84%	0.2961s	0.0487s	10.3115s
Weighted Ensemble L2	99.71%	100.0%	2.4196s	0.5423s	220.5215s
XGBoost	99.65%	99.80%	0.2705s	0.0319s	7.1429s
Neural Network Fast AI	99.54%	99.84%	0.3548s	0.0736s	169.7237s
Cat Boost	99.54%	99.68%	0.0672s	0.0129s	65.0674s
Neural Network PyTorch	99.51%	99.64%	0.1500s	0.0316s	158.1814s
Light GBMXT	99.43%	99.72%	0.5668s	0.0653s	9.6176s
K-Neighbors Distance	98.61%	98.36%	15.7458s	3.4580s	0.2718s
K-Neighbors Uniform	98.40%	98.20%	9.6780s	3.2489s	3.9128s

D. Conclusion

The article focused on using neural network-based imputation techniques and feature selection using bat algorithm in network attack detection. The algorithm's comprehensive exploration capabilities make it well-suited for optimizing feature selection. When integrated with the neural network classifier, the model achieved impressive results, converging with a 99.4% accuracy and maintaining high average F_1 score, recall, and precision values of 97%, 97%, and 98%, respectively. The neural network-based imputation technique demonstrates outstanding performance,

achieving accuracy rates of 99.4% on validation data and 99.3% on test data. This method consistently maintains precision, recall, and F_1 score around 98%. Models using this method also approach perfection in classifying normal and neptune labels. This imputation method can also be applied to other model architectures using autoML. Alternative models such as Light GBM, XG Boost, Random Forest, Extra Trees, and Weighted Ensemble L2 also exhibit exceptional accuracy, exceeding 99.8%. However, the training process using the neural network architecture and Weighted Ensemble L2 takes a longer time compared to other architectures but its prediction speed is not significantly worse when compared to other models.

E. References

[1] V. Fransisca and W. Ningsih, "The Advancement of Technology and its Impact on Social Life in Indonesia," *Devot. J. Res. Community Serv.*, vol. 4, no. 3, pp. 860–864, Mar. 2023, doi: 10.36418/devotion.v4i3.445.

[2] N. K. N. Widiasari and E. F. Thalib, "The Impact of Information Technology Development on Cybercrime Rate in Indonesia," *J. Digit. Law Policy*, vol. 1, no. 2, pp. 73–86, Jan. 2022, doi: 10.58982/jdlp.v1i2.165.

[3] G. W. McDiarmid and Y. Zhao (赵勇), "Time to Rethink: Educating for a Technology-Transformed World," *ECNU Rev. Educ.*, vol. 6, no. 2, pp. 189–214, May 2023, doi: 10.1177/20965311221076493.

[4] Masruddin, "The Importance of Using Technology in English Teaching and Learning," *IDEAS J. Engl. Lang. Teach. Learn. Linguist. Lit.*, vol. 2, no. 2, Dec. 2014, doi: https://doi.org/10.24256/ideas.v2i2.36.

[5] S. K. R., "Technology and Transformation in Communication," *J. Adv. Res. Electr. Electron. Eng. ISSN 2208-2395*, vol. 5, no. 8, pp. 01–13, Aug. 2018, doi: 10.53555/nneee.v5i8.157.

[6] M. Stone, "The evolution of the telecommunications industry — What can we learn from it?," *J. Direct Data Digit. Mark. Pract.*, vol. 16, no. 3, pp. 157–165, Jan. 2015, doi: 10.1057/dddmp.2014.80.

[7] A. R. Hakim, "Profile and Role of Telecommunication Sector in Indonesia," 2011, *Unpublished*. doi: 10.13140/RG.2.1.4514.9680.

[8] Naeem. A. Nawaz, A. Abid, S. Rasheed, M. S. Farooq, A. Shahzadi, and I. Mubarik, "Impact of Telecommunication Network on Future of Telemedicine in Healthcare: A systematic Literature Review," *Int. J. Adv. Appl. Sci.*, vol. 9, no. 7, pp. 122–138, Jul. 2022, doi: 10.21833/ijaas.2022.07.013.

[9] Y. Li and Q. Liu, "A Comprehensive Review Study of Cyber-Attacks and Cyber Security: Emerging Trends and Recent Developments," *Energy Rep.*, vol. 7, pp. 8176–8186, Nov. 2021, doi: 10.1016/j.egyr.2021.08.126.

[10] N. S.Mangrulkar, A. R. Bhagat Patil, and A. S. Pande, "Network Attacks and Their Detection Mechanisms: A Review," *Int. J. Comput. Appl.*, vol. 90, no. 9, pp. 37–39, Mar. 2014, doi: 10.5120/15606-3154.

[11] R. M. A. Haseeb-ur-rehman *et al.*, "High-Speed Network DDoS Attack Detection: A Survey," *Sensors*, vol. 23, no. 15, p. 6850, Aug. 2023, doi: 10.3390/s23156850.

[12] S. Anwar *et al.*, "From Intrusion Detection to an Intrusion Response System: Fundamentals, Requirements, and Future Directions," *Algorithms*, vol. 10, no. 2, p. 39, Mar. 2017, doi: 10.3390/a10020039.

[13] Y. Wu, D. Wei, and J. Feng, "Network Attacks Detection Methods Based on Deep Learning Techniques: A Survey," *Secur. Commun. Netw.*, vol. 2020, pp. 1–17, Aug. 2020, doi: 10.1155/2020/8872923.
[14] K. S. Niraja and S. Srinivasa Rao, "WITHDRAWN: A hybrid algorithm design for near real time detection cyber attacks from compromised devices to enhance IoT security," *Mater. Today Proc.*, p. S2214785321008488, Mar. 2021, doi: 10.1016/j.matpr.2021.01.751.
[15] M. Aljabri *et al.*, "Intelligent Techniques for Detecting Network Attacks: Review and Research Directions," *Sensors*, vol. 21, no. 21, p. 7070, Oct. 2021, doi: 10.3390/s21217070.
[16] N. Wattanapongsakorn *et al.*, "A Practical Network-Based Intrusion Detection and Prevention System," in *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*, Liverpool, United Kingdom: IEEE, Jun. 2012, pp. 209–214. doi: 10.1109/TrustCom.2012.46.
[17] M. Sohail *et al.*, "Triangle Area Based Multivariate Correlation Analysis for Detecting and Mitigating Cache Pollution Attacks in Named Data Networking," in *2020 3rd International Conference on Hot Information-Centric Networking (HotICN)*, Hefei, China: IEEE, Dec. 2020, pp. 114–121. doi: 10.1109/HotICN50779.2020.9350746.
[18] A. Kumaravel and M. Niraisha, "Multi-Classification Approach for Detecting Network Attacks," in *2013 IEEE CONFERENCE ON INFORMATION AND COMMUNICATION TECHNOLOGIES*, Thuckalay, Tamil Nadu, India: IEEE, Apr. 2013, pp. 1114–1117. doi: 10.1109/CICT.2013.6558266.
[19] K. A. Dhanya, S. Vajipayajula, K. Srinivasan, A. Tibrewal, T. S. Kumar, and T. G. Kumar, "Detection of Network Attacks using Machine Learning and Deep Learning Models," *Procedia Comput. Sci.*, vol. 218, pp. 57–66, 2023, doi: 10.1016/j.procs.2022.12.401.
[20] G. Teshome, "Development of a Method for Detecting Network Attack using Machine learning Algorithms | Request PDF," *SSRN Electron. J.*, vol. 5, no. 56, Nov. 2022, doi: 10.2139/ssrn.4106481.
[21] M. M. Taye, "Understanding of Machine Learning with Deep Learning: Architectures, Workflow, Applications and Future Directions," *Computers*, vol. 12, no. 5, p. 91, Apr. 2023, doi: 10.3390/computers12050091.
[22] H. Mamdouh Farghaly and T. Abd El-Hafeez, "A High-Quality Feature Selection Method Based on Frequent and Correlated Items for Text Classification," *Soft Comput.*, vol. 27, no. 16, pp. 11259–11274, Aug. 2023, doi: 10.1007/s00500-023-08587-x.
[23] B. Pes, "Ensemble Feature Selection for High-Dimensional Data: A Stability Analysis Across Multiple Domains," *Neural Comput. Appl.*, vol. 32, no. 10, pp. 5951–5973, May 2020, doi: 10.1007/s00521-019-04082-3.
[24] M. Afshar and H. Usefi, "Optimizing Feature Selection Methods by Removing Irrelevant Features Using Sparse Least Squares," *Expert Syst. Appl.*, vol. 200, p. 116928, Aug. 2022, doi: 10.1016/j.eswa.2022.116928.
[25] N. Pudjihartono, T. Fadason, A. W. Kempa-Liehr, and J. M. O'Sullivan, "A Review of Feature Selection Methods for Machine Learning-Based Disease Risk Prediction," *Front. Bioinforma.*, vol. 2, p. 927312, Jun. 2022, doi: 10.3389/fbinf.2022.927312.

[26] R. Ge *et al.*, "McTwo: a Two-step Feature Selection Algorithm Based on Maximal Information Coefficient," *BMC Bioinformatics*, vol. 17, no. 1, p. 142, Mar. 2016, doi: 10.1186/s12859-016-0990-0.
[27] T. Verdonck, B. Baesens, M. Óskarsdóttir, and S. Vanden Broucke, "Special Issue on Feature Engineering Editorial," *Mach. Learn.*, vol. 113, no. 7, pp. 3917–3928, Jul. 2024, doi: 10.1007/s10994-021-06042-2.
[28] T. Islam, M. E. Islam, and M. R. Ruhin, "An Analysis of Foraging and Echolocation Behavior of Swarm Intelligence Algorithms in Optimization: ACO, BCO and BA," *Int. J. Intell. Sci.*, vol. 08, no. 01, pp. 1–27, 2018, doi: 10.4236/ijis.2018.81001.
[29] A. Chakri, R. Khelif, M. Benouaret, and X.-S. Yang, "New Directional Dat Algorithm for Continuous Optimization Problems," *Expert Syst. Appl.*, vol. 69, pp. 159–175, Mar. 2017, doi: 10.1016/j.eswa.2016.10.050.
[30] A. R. Ismail, N. Z. Abidin, and M. K. Maen, "Systematic Review on Missing Data Imputation Techniques with Machine Learning Algorithms for Healthcare," *J. Robot. Control JRC*, vol. 3, no. 2, pp. 143–152, Feb. 2022, doi: 10.18196/jrc.v3i2.13133.
[31] X. Xu, L. Xia, Q. Zhang, S. Wu, M. Wu, and H. Liu, "The ability of Different Imputation Methods for Missing Values in Mental Measurement Questionnaires," *BMC Med. Res. Methodol.*, vol. 20, no. 1, p. 42, Dec. 2020, doi: 10.1186/s12874-020-00932-0.
[32] P. J. Rousseeuw and M. Hubert, "Anomaly Detection by Robust Statistics," *WIREs Data Min. Knowl. Discov.*, vol. 8, no. 2, p. e1236, Mar. 2018, doi: 10.1002/widm.1236.
[33] S. J. Choudhury and N. R. Pal, "Imputation of Missing Data with Neural Networks for Classification," *Knowl.-Based Syst.*, vol. 182, p. 104838, Oct. 2019, doi: 10.1016/j.knosys.2019.07.009.
[34] X.-S. Yang, "A New Metaheuristic Bat-Inspired Algorithm," in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, vol. 284, J. R. González, D. A. Pelta, C. Cruz, G. Terrazas, and N. Krasnogor, Eds., in Studies in Computational Intelligence, vol. 284. , Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 65–74. doi: 10.1007/978-3-642-12538-6_6.
[35] R. Y. M. Nakamura, L. A. M. Pereira, K. A. Costa, D. Rodrigues, J. P. Papa, and X.-S. Yang, "BBA: A Binary Bat Algorithm for Feature Selection," in *2012 25th SIBGRAPI Conference on Graphics, Patterns and Images*, Ouro Preto, Brazil: IEEE, Aug. 2012, pp. 291–297. doi: 10.1109/SIBGRAPI.2012.47.