2022 7th International Conference on Business and Industrial Research (ICBIR) | 978-1-6654-9475-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICBIR54589.2022.9786429

Anomaly Detection based on NSL-KDD using XGBoost with Optuna Tuning

Farah Hana Kusumaputri
Department of Engineering
Universitas Indonesia
Depok, Indonesia
farah.hana@ui.ac.id

Ajib Setyo Arifin
Department of Engineering
Universitas Indonesia
Depok, Indonesia
ajib@eng.ui.ac.id

***Abstract*—The enormous internet development now day across all aspects of human life has introduced various hidden risk of malicious attacks on network security that most users didn't realize. One of the malicious attacks is intrusion of system that proliferate user's account effortlessly. Hence, in order to avoid intrusion effect that lead to financial loss and any other loss, intrusion detection system is needed to identify a dynamic pattern of cyber attacks. In this paper, we propose an Optimized XGBoost Classifier model with the help of Optuna Hypertuning method to find the best parameter for the model. In order to find the most efficient method for training, we assign three Optuna scenarios combine with feature selection to learn the data and the machine learning model. Through learning, Optuna generated the best parameter for XGBoost Classifier. Optuna avoids time consuming and low efficiency training model. The propose XGBoost Classifier model with Optuna Hypertuning method results in a greater accuracy of detection intrusion compare to any other models.**

***Keywords*-Intrusion, NSL-KDD, XGBoost, Optuna, Machine Learning**

I. Introduction

Computer networks have become one of the most segment keys in many aspects of daily lives, from field of the business sector, lecturing data process, and entertainment. The rapid development of technology provides the existence of various ease of access to the internet. However, the ease use of internet access that is exponentially increase make the emerge a new threat in the form of intrusion. Intrusion is the act of entering a person's location or network without clear permission or the act of intruders to infiltrate someone's server to gain some advantages [1]. In recent years, especially in Indonesia, intrusion has become serious issue since costumer's data of one of the biggest unicorn company in Indonesia leaked in black market, following by the latest case of medical record of Indonesian patients on popular platform for tracking COVID-19 in Indonesia being sold in black market. These matters potentially cause losses especially in user's data that was successfully stolen, it might be used for any unwanted crimes, also on computer networks that were successfully burglarized into, therefore a robust detection system is needed in order to detect abnormalities in a computer's network traffic. Intrusion Detection System (IDS) is one of the security management methods that can be used to monitor computer networks, as well as detect anomalies that occur in traffic, so that intrusion can be prevented and user data can be protected properly [1]. Sometimes, in one situation, a server can be visited by many clients, so that it normally generates large traffic [2], hence a robust IDS that is able to detect abnormalities in large network traffic which aim preventing data – lost data as well as other losses that may occur in the future. In this paper, we propose a robust IDS in order to detect anomalies on data traffic, by using XGBoost combined with pre-processing method using feature selection on (Network Security Layer – Knowledge Discovery in Database) NSL-KDD dataset and optimized by Optuna tuning.

II. Related Works

Several studies have been carried out on IDS, one of them is from Bhupendra Ingre and Anamika Yadav which using Artificial Neural Network (ANN) for evaluating the NSL-KDD dataset [1]. The dataset are divided into four category attacks symbolized in binary form and have additional 17 type of attacks in testing performance. This work proposed Levenberg-Marquardt (LM) and Broydon-Fletcher-Goldfarb-Shanno (BFGS) Quasi-Newton Backpropagation algorithm for processing the dataset. The result showed that this method has 81.2% accuracy for binary classification which classifying the system whether they are getting attack by one type of attack (DoS, Probe, R2L, and U2R) or not and for multiclass classification (DoS, Probe, R2L, and U2R and normal condition), this work has 79.9% accuracy.

Another study from Liu and Song proposes new pre-processing method by combining the improved smote algorithm for up-sampling with small amount of data while also down-sampling a large amount of data in order to make the dataset balance, there are also one-hot encoding method and standardization data for making the dataset turns into numerical form in the range of [0,1] [3]. After the pre-processing, KDD Cup '99 dataset is being processed by using XGBoost classifier, it turn out that this method successfully outperforms any mentioned method by the average precision of 79.4%.

A new dataset, UNSW-NB15 is used in to IDS [4]. In that paper, authors used the dataset in order to provide comprehensive and representative to modern network. This dataset was processed using XGBoost classifier, compared to other existing methods. XGBoost classifier successfully gained the highest validation accuracy, 88%.

Support Vector Machine (SVM) is also deployed on NSL-KDD for making robust IDS [5]. NSL-KDD does not include redundant training instances, so it will not confuse the system. After the system proposed, several parameters such as true positive, true negative, false positive and false negative are used to compute the performance of the system. After validation conducted, the highest validation accuracy comes from 41 features by 82.37%.

Another research conducted by Preethi Devan and Neelu Khare are using XGBoost technique followed by Deep Neural Network (DNN) for classification of network intrusion [6]. At first, min-max normalization is used in order to make all the features have the same range, which help the model train well in numeric form. Feature selection is also deployed to help the system learns efficiently, after the training and validation process occur by deploying DNN optimized by Adam optimizer, the result shows promising accuracy compared to others by getting 97% consistently.

NSL-KDD dataset is also use in [2]. The raw data firstly get normalization range between 0—1, after that, classification is processed by using J48, SVM and Naïve Bayes algorithm. After several iterations by adding Correlation based Feature Selection (CFS) for dimensional reduction, J48 has the highest accuracy in detecting six features of attack, namely normal, DoS, Probe, U2R, and R2L by the average accuracy about 98%.

Another effective IDS is also built by using XGBoost [7]. This work also uses NSL-KDD as the experimental dataset. At the first is regularization of dataset by converting into numeric version. This experiment also uses confusion matrix to get accuracy, true positive rate and true negative rate. Based on result, the highest accuracy gain is 98.70%.

A combination between XGBoost and Whale Optimization Algorithm (WOA) is also used and [8] combined with pre-processing Principal Component Analysis (PCA) for dimensional reduction, to solve the issue of unbalanced dataset. After this process, the XGBoost classifier classifies the dataset combine with WOA to determine the best parameter to and boost the XGBoost accuracy. Compared to other methods, this method sucessfuly outperform others by having 99.06% accuracy.

The use of NSL-KDD dataset is also utilized in [9]. The experiment is about having the IDS by using deep learning approach system to train itself with the patterns of abnormalities happen on network by deploying sparse auto-encoders and logistic regression classifier. Pre-processing also implies in this system in order to make normalization range and make all the dataset on numeric form. After several iteration underway, the overall model accuracy gain is 87.2%.

Other experiment uses XGBooster combine with particle swarm optimization (PSO) [10]. PSO has main role to optimize six parameters that have great influence on the model, they are *learning data*, *maximum tree depth*, *minimum leaf weight*, *gamma, sub-sample and colsample_bytree*, also used for adaptively search for optimal structure of the proposed method. The experiment that uses NSL-KDD dataset showed PSO-XGBooster successfully outperform any other comparative models in precision, recall, macro-average and mean average precision especially when identifying minority attacks such as U2R and R2L. This research uses macro curves for evaluating the average precision since the dataset is imbalance, the highest average precision is 92%.

III. Proposed Methods

This research proposed a gradient boosting based machine learning model as a network anomaly detector. We used XGBoost tuned with Optuna hyperparameter-tuning (or hypertuning) to increase detection quality. The proposed network anomaly detection model was trained and tested using data provided by NSL-KDD dataset. Finally, the machine learning anomaly detection output's quality metered using accuracy, recall, precision, and F1 score values. The larger those values, the better those results. The scheme of proposed method can be seen in Figure 1.

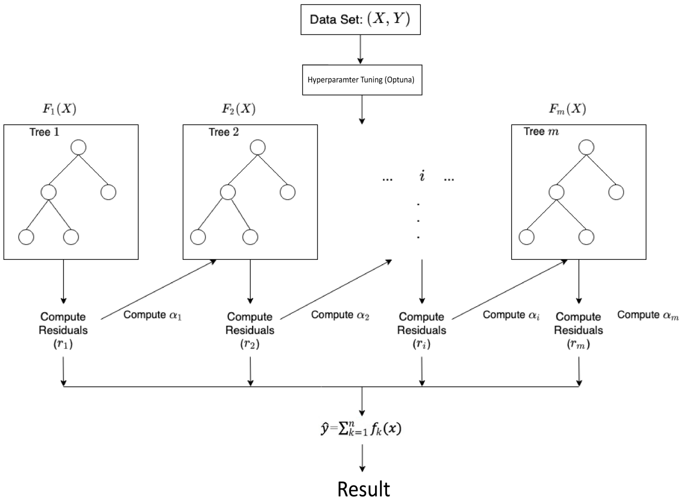
Data Set: (X, Y)
Hyperparamter Tuning (Optuna)
$F_1(X)$
$F_2(X)$
$F_m(X)$
Tree 1
Tree 2
Tree m
... i ...
Compute Residuals (r_1)
Compute α_1
Compute Residuals (r_2)
Compute α_2
Compute Residuals (r_i)
Compute α_i
Compute Residuals (r_m)
Compute α_m
$\hat{y}=\sum_{k=1}^{n} f_k(x)$
Result

Fig. 1. Scheme of Proposed Method

TABLE I. Features in NSL-KDD Dataset

S. No	Feature Name	S. No	Feature Name
1	Duration	23	Count
2	Protocol_type	24	Srv_count
3	Service	25	Serror_rate
4	Flag	26	Srv_serror_rate
5	Src_bytes	27	Rerror_rate
6	Dst_bytes	28	Srv_rerror_rate
7	Land	29	Same_srv_rate
8	Wrong_fragment	30	Diff_srv_rate
9	Urgent	31	Srv_diff_host_rate
10	Host	32	Dst_host_count
11	Num_failed_logins	33	Dst_host_srv_count
12	Logged_in	34	Dst_host_same_srv_rate
13	Num_compromised	35	Dst_host_diff_srv_rate
14	Root_shell	36	Dst_host_same_src_port_rate
15	Su_attemped	37	Dst_host_srv_diff_host_rate
16	Num_root	38	Dst_host_serror_rate
17	Num_file_creations	39	Dst_host_srv_serror_rate
18	Num_shells	40	Dst_host_rerror_rate
19	Num_access_files	41	Dst_host_rerror_rate
20	Num_outbound_cmds		
21	Is_host_login		
22	Is_guest_login		

A. NSL-KDD Dataset

NSL-KDD is a the dataset that contains internet traffic records. This dataset has become an effective and preferred benchmark dataset to compare several network intrusion detection methods. NSL-KDD is the modified version of KDD Cup '99 dataset [11]. The KDD Cup '99 has many duplicated and redundant records, which cause other record classifications to be blocked. After all, KDD Cup '99 is not like an accurate simulation of real network traffic. Meanwhile, the NSL-KDD dataset aims to overcome these problems. The NSL-KDD dataset consists of a reasonably selected feature number from the KDD Cup '99, including redundancy eliminated on the training and testing data. The classification performance result is not bias [12]. The name of every feature on the NSL-KDD dataset can be seen in Table I.

B. Features Selection

Feature selection is a pre-processing method which use for detecting the relevant features for proposed model and discarding the rest features in order to make the model run efficiently [13]. This feature technique use wrapper, embedded, and filter feature selection techniques in order to make the selected ones has significant. Filter techniques select which features is independent among any features. Wrapper models utilize the classifier to evaluate on it and find the optimal features for proposed model. Embedded techniques search the optimal feature subset during the model building process [13].

This feature selection is also supported by extra tree, which can select most predominant features effectively, so it will reduce computational execution time. In this process, each tree is provided with a random sample of NSL-KDD features. The feature selection will be performed by ordering the random features in descending order based [14]. In this paper we used the Extra Tree Classifier for the feature importance selection process to implementing on 10 features, 20 features, 30 features, and 40 features, to choose which is the best proposed model on this paper.

C. XGBoost

XGBoost is an adaptable machine learning system for optimized tree boosting and also is available as open-source packages. XGBoost regularized model is used to prevent the model from overfitting, and simplifies function for pararelization of the regularized greedy forest algorithm [15]. Furthermore, XGBoost is very tolerant of missing value with learning automatically dealing with missing values. The XGBoost supports customized object function and evaluation function. It will take several types of input data features, unlike other models that usually only process one kind of datatype. It is proven in several different dataset resulted better performance [16].

XGBoost was recently proposed by Chen and Guestrinis [17]. Based on the original framework of gradient boosting, it uses K additive trees to approximate the output $\hat{y}_i$ in the following equation (1)[18]:

$$\hat{y}_i = \sum_{k=1}^{K} f_k\left(X_i\right), f_k \in F \tag{1}$$

Where f_k is an independent Classification and Regression Tree (CART) at the each of k steps which it will be maps the input variables X_i to $\hat{y}_i$, and the F is the space of functions containing all CART. The different between the XGBoost and the original gradient boosting algorithm, XGBoost aims at minimizing the regularized object function defined in equation (2):

$$Obj = \sum_{i=1}^{n} \iota\left(y_i, \hat{y}_i\right) + \sum_{k=1}^{K} \Omega\left(f_k\right), f_k \in F \tag{2}$$

Where $\Omega(f_k) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$. The regularized objective function contains two parts: the training loss function $_l$ and the regularization term of Ω. The training loss $_l$ measures the difference between the predicted value $\hat{y}_i$ and the true value y_i. The regularization term Ω, measures the complexity of model, which helps to smooth the final learnt weight to avoid overfitting [18].

D. Optuna Tuning

Optuna is an automation software framework designed to enable the tuning of hyperparameter models. Optuna searches the ideal hyperparameter value for the model that has been chosen to utilize various samplers, for grid search, random samplers, and genetic calculations. Optuna presents each interaction as a study which the improvements depend on the objective trial and function. The context of Optuna resorts to an architecture that enables the user to construct the search space dynamically. The utilization of the methods and trial object during the run time of the objective function is resulted in the dynamic construction of the search space [19].

Hyperparameter values are optimized by the formalization of Optuna that returns the validation score from the process of minimizing or maximizing an objective function to which takes hyperparameters as an input. The efficiency of the dynamic searching strategy has a critical function of determining the set of hyperparameters and performance estimation that estimates the value of the set of hyperparameters returned from the discarded learning curves and hyperparameters. The dynamic effectiveness of both searching strategy and performance estimation strategy is required on building the most cost-effective solution to the optimization of the model [20].

E. Result Evaluations

The performance measure of machine learning methods can be evaluated using Confusion Matrix parameters. The standard confusion matrix parameters in this paper include precision accuracy, recall accuracy, and F1-Score accuracy, with given parameters, were calculated using True Positive (TP), True Negative, False Negative (FN), and False Positive (FP). The confusion matrix is shown in Table 2.

TABLE II. Confussion Matrix

		Predicted	
		Yes	No
Actual	Yes	True Positive (TP)	False Negative (FN)
	No	False Positive (FP)	True Negative (TN)

By using the weighted precision values, weighted recall values, and weighted F1-Score values to evaluate the system performances. The average weight was calculated by multiplying the average precision, F1-score, and recall for each class, divided by the amount of the data for each class. Equations (3), (4), (5) and (6) shows the calculation of accuracy, weighted precision, recall, and F1-score.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{3}$$

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

$$F1 - Score = \frac{Precision.Recall}{Precision + Recall} \tag{6}$$

IV. Experimental Result

A. Dataset

In total, there are 42 features on NSL-KDD dataset [1], both available in training and testing data. Features details shown in Table I. In the first condition NSL-KDD datasets has seperated the data between training and test, so in this paper, there's no splitting data in the process for both training and testing.

B. Feature Selection using Feature Importances

In this paper, Feature importance using Extra Tree Classifier were employed to selecting the features, there are several features for result model comparison using XGBoost which going to be used: 10 features, 20 features, 30 features, and 40 features.

C. Training and Validation Process

There are three Optuna Hyperparameter tuning methods that use for getting the best model. The Optuna Hyperparameter tuning differs in the hyperparameter that is tuned.

TABLE III. Tuning on Optuna Hyperparameters

Scenarios	Hyperparameter
Optuna Hyperparameter Scenario 1	learning_rate reg_lambda reg_alpha subsample colsample_bytree max_depth
Optuna Hyperparameter Scenario 2	num_leaves n_estimators max_depth min_child_samples learning_rate min_data_in_leaf bagging_fraction feature_fraction
Optuna Hyperparameter Scenario 3	learning_rate reg_lambda reg_alpha subsample colsample_bytree num_leaves n_estimators max_depth min_child_samples min_data_in_leaf bagging_fraction feature_fraction

For the first Optuna scenario, the hyperparameter that is used are *learning_rate, reg_lambda, reg_alpha, subsample, colsample_bytree,* and *max_depth*. The hyperparameters for the second Optuna scenario are *num_leaves, n_estimators, max_depth, min_child_samples, learning_rate, min_data_in_leaf, bagging_fraction,* and *feature_fraction*. The hyperparameters for the last Optuna tuning is the composition of the first two scenarios, they

[1] Dataset can be accessed in https://www.kaggle.com/hassan06/nslkdd

are *learning_rate, reg_lambda, reg_alpha, subsample, colsample_bytree, num_leaves, n_estimators, max_depth, min_child_samples, min_data_in_leaf, bagging_fraction,* and *feature_fraction*. In brief, each Optuna Hyperparameter tuning scenario is explained in Table III.

After several experiment, the results show that XGBoost with 40 feature selections combined with Optuna hyperparameter tuning scenario 2 shows the best result, any other result shown in Table IV.

TABLE IV. Experimental Result

Model	Optuna Scenario	Accruracy	F1-Score	Recall	Precision
XGBoost with 40 Feature Selection	No Optuna	0.84204	0.84301	0.74503	0.97065
	Scenario 1	0.98074	0.98333	0.99800	0.96909
	Scenario 2	**0.99563**	**0.99617**	0.99822	0.98161
	Scenario 3	0.99295	0.99384	0.99850	0.98415
XGBoost with 30 Feature Selection	No Optuna	0.79471	0.78548	0.66025	0.96933
	Scenario 1	0.93658	0.94558	0.96797	0.92423
	Scenario 2	0.98509	0.98692	0.98843	0.95483
	Scenario 3	0.95132	0.95814	0.97868	0.94937
XGBoost with 20 Feature Selection	No Optuna	0.77794	0.76348	0.62962	096963
	Scenario 1	0.86192	0.88821	0.96314	0.82439
	Scenario 2	0.95873	0.96431	0.97938	0.88706
	Scenario 3	0.94513	0.95323	0.98226	0.900001
XGBoost with 10 Feature Selection	No Optuna	0.43111	0.00124	0.00062	1.0
	Scenario 1	0.92236	0.93386	0.96314	0.90635
	Scenario 2	0.95458	0.96065	0.97353	0.92726
	Scenario 3	0.93711	0.94640	0.97520	0.9246

V. Conclusion

To summarize, the best solution is by having 40 feature selections and using Optuna Hyperparameter Scenario 2 to tune the hyperparameters of the XGBoost intrusion detection model. By tuning hyperparameters such as *num_leaves, n_estimators, max_depth, min_child_samples, learning_rate, min_data_in_leaf, bagging_fraction,* and *feature_fraction* using Optuna, researcher able to outperform the existing model. The proposed intrusion detection model resulted 0.995635 in accuracy, 0.996174 in F1-Score, 0.998223 in Recall, and 0.981612 Precision. The results outperform the XGBoost Classifier with No Optuna by 0.52 in accuracy and 0.24 by Recall.

References

[1] B. Ingre and A. Yadav, "Performance analysis of nsl-kdd dataset using ann," in *2015 international conference on signal processing and communication engineering systems*. IEEE, 2015, pp. 92–96. [Online]. Available: http://dx.doi.org/10.1109/SPACES.2015.7058223

[2] L. Dhanabal and S. Shantharajah, "A study on nsl-kdd dataset for intrusion detection system based on classification algorithms," *International journal of advanced research in computer and communication engineering*, vol. 4, no. 6, pp. 446–452, 2015. [Online]. Available: http://dx.doi.org/10.17148/IJARCCE.2015.4696

[3] P. Su, Y. Liu, and X. Song, "Research on intrusion detection method based on improved smote and xgboost," ser. ICCNS 2018. New York, NY, USA: Association for Computing Machinery, 2018, p. 37–41. [Online]. Available: https://doi.org/10.1145/3290480.3290505

[4] A. Husain, A. Salem, C. Jim, and G. Dimitoglou, "Development of an efficient network intrusion detection model using extreme gradient boosting (xgboost) on the unsw-nb15 dataset," in *2019 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, 2019, pp. 1–7. [Online]. Available: https://doi.org/10.1109/ISSPIT47144.2019.9001867

[5] M. S. Pervez and D. M. Farid, "Feature selection and intrusion classification in nsl-kdd cup 99 dataset employing svms," in *The 8th International Conference on Software, Knowledge, Information Management and Applications (SKIMA 2014)*, 2014, pp. 1–6. [Online]. Available: https//doi,org/10.1109/SKIMA.2014.7083539

[6] P. Devan and N. Khare, "An efficient xgboost–dnn-based classification model for network intrusion detection system," *Neural Computing Applications*, vol. 32, pp. 12 499—-12 514, 2020. [Online]. Available: https://doi.org/10.1007/s00521-020-04708-x

[7] S. S. Dhaliwal, A.-A. Nahid, and R. Abbas, "Effective intrusion detection system using xgboost," *Information*, vol. 9, no. 7, p. 149, 2018. [Online]. Available: https://doi.org/10.3390/info9070149

[8] P. X. Yan Song, Haowei Li and D. Liu, "A method of intrusion detection based on woa-xgboost algorithm," *Discrete Dynamics in Nature and Society*, vol. 2022, 2022. [Online]. Available: https://doi.org/10.1155/2022/5245622

[9] S. Gurung, M. K. Ghose, and A. Subedi, "Deep learning approach on network intrusion detection system using nsl-kdd dataset," *International Journal of Computer Network and Information Security*, vol. 11, no. 3, pp. 8–14, 2019. [Online]. Available: https://doi.org/10.5815/ijcnis.2019.03.02

[10] H. Jiang, Z. He, G. Ye, and H. Zhang, "Network intrusion detection based on pso-xgboost model," *IEEE Access*, vol. 8, pp. 58 392–58 401, 2020. [Online]. Available: http://doi.org/10.1109/ACCESS.2020.2982418.

[11] P. Devan and N. Khare, "An efficient xgboost–dnn-based classification model for network intrusion detection system," *Neural Computing and Applications*, pp. 1–16, 2020. [Online]. Available: https://doi.org/10.1007/s00521-020-04708-x

[12] D. D. Proti'c, "Review of kdd cup '99, nsl-kdd and kyoto 2006+ datasets," *Vojnotehniki Glasnik*, vol. 66, pp. 580–596, 2018. [Online]. Available: https://doi.org/10.5937/vojtehg66-16670

[13] M. Saarela and S. Jauhiainen, "Comparison of feature importance measures as explanations for classification models," *SN Applied Sciences*, vol. 3, no. 2, pp. 1–12, 2021. [Online]. Available: https://doi.org/10.1007/s42452-021-04148-9

[14] "Leukocyte classification based on spatial and spectral features of microscopic hyperspectral images," *Optics Laser Technology*, vol. 112, pp. 530–538, 2019.

[15] J. Henriques, F. Caldeira, T. Cruz, and P. Simões, "Combining k-means and xgboost models for anomaly detection using log datasets," *Electronics*, vol. 9, no. 7, 2020. [Online]. Available: https://doi.org/10.3390/electronics9071164

[16] V. U. Pugliese, C. M. Hirata, and R. D. Costa, "Comparing supervised classification methods for financial domain problems." in *ICEIS (1)*, 2020, pp. 440–451. [Online]. Available: https://doi,org/10.5220/0009426204400451

[17] T. Chen and C. Guestrin, "Xgboost," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Aug 2016. [Online]. Available: http://dx.doi.org/10.1145/2939672.2939785

[18] Y. Meng, N. Yang, Z. Qian, and G. Zhang, "What makes an online review more helpful: An interpretation framework using xgboost and shap values," *Journal of Theoretical and Applied Electronic Commerce Research*, vol. 16, no. 3, p. 466–490, Nov 2020. [Online]. Available: https://doi.org/10.3390/jtaer16030029

[19] S. Garg and P. Pundir, "Mofit: A framework to reduce obesity using machine learning and iot," in *2021 44th International Convention on Information, Communication and Electronic Technology (MIPRO)*, 2021, pp. 1733–1740. [Online]. Available: https://doi,org/10.23919/MIPRO52101.2021.9596673

[20] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, ser. KDD '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 2623–2631. [Online]. Available: https://doi.org/10.1145/3292500.3330701