Modification of a Density-Based Spatial Clustering Algorithm for Applications with Noise for Data Reduction in Intrusion Detection Systems

Wiharto, Aditya K. Wicaksana, and Denis E. Cahyani

Department of Informatic, Universitas Sebelas Maret, Surakarta, Indonesia

ljfis

Abstract

Monitoring activity in computer networks is required to detect anomalous activities. This monitoring model is known as an intrusion detection system (IDS). Most IDS model developments are based on machine learning. The development of this model requires activity data in the network, either normal or anomalous, in sufficient amounts. The amount of available data also has an impact on the slow learning process in the IDS system, with the resulting performance sometimes not being proportional to the amount of data. This study proposes an IDS model that combines DBSCAN modification with the CART algorithm. DBSCAN modification is performed to reduce data by adding a *MinNeighborhood* parameter, which is used to determine the distance of the density to the cluster center point, which will then be marked for deletion. The test results, using the Kaggle and KDDCup99 datasets, show that the proposed system model is able to maintain a classification accuracy above 90% for 80% data reduction. This performance was also followed by a decrease in computation time, for the Kaggle dataset from 91.8 ms to 31.1 ms, while for the KDDCup99 dataset from 5.535 seconds to 1.120 seconds.

Keywords: Clustering, CART, Intrusion detection system, DBSCAN, Data reduction

1. Introduction

The growth of the Internet and computer network poses a serious threat to data security [1]. Various gaps in security systems are increasingly being found along with the development of Internet technology [2]. To maintain data integrity and confidentiality, the creation of a system that can detect attacks and log any anomaly in a network is of great importance [3]. One of the efforts to prevent attacks from occurring again is to use an intrusion detection system (IDS). An IDS works by detecting based on the search for the specific characteristics of the attempted attack. There are also IDSs that work by detecting based on a comparison of normal traffic patterns with abnormal or anomalous traffic patterns. Abnormal traffic patterns, such as DoS, U2R, probe, and R2L [4]. Various techniques for intrusion detection and performance parameter accuracy are important parameters in an IDS. The detection rate and false alarm rate, in particular, play important roles in the accuracy analysis. Intrusion detection should be enriched to reduce false alarms and increase detection speed [5, 6].

IDS is mostly developed based on machine learning [5–7]. This IDS classifies every activity

Received: Feb. 9, 2021

Revised : May 12, 2021

Accepted: Jun. 9, 2021

Correspondence to: Wiharto

(wiharto@staff.uns.ac.id)

©The Korean Institute of Intelligent Systems

©This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (<http://creativecommons.org/licenses/by-nc/3.0/>) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

in the network, whether normal or anomalous. This classification system serves to study and predict attack patterns [8]. In the classification, to create a model, a data pattern learning process called learning is needed. The learning process, for classification, requires time proportional to the amount of data being processed. Thus, the greater the amount of data processed, the longer the computation time and the higher the resources required. In a learning system, the more data that is used, the smaller the error rate (i.e., the accuracy increases) [9, 10]; however, the large amount of data that is not well distributed also results in high computation time, which is not proportional to the accuracy performance parameters it produces.

A large amount of data and the number of attributes, known as the data dimension, will influence the learning process. Large data dimensions can also worsen system performance, depending on the data distribution model and the type of attributes used. These conditions require machine learning for the dimensional reduction process. Dimension reduction can reduce the number of data or reduce the number of attributes, or both [11]. The data reduction process can be performed using clustering techniques, which aim to cluster data into small clusters to be used for learning, as was done by Wiharto et al. [12]. In this study, clustering was carried out, and then each cluster was used for the Levenberg-Marquardt learning algorithm and the quasi-Newton algorithm, to be combined with the naïve Bayesian algorithm for the conclusion. The use of clustering for data reduction was also carried out by Uhm et al. [11], which combines data reduction and attribute reduction.

The use of clustering in the research of Uhm et al. [11] and Wiharto et al. [12] aims to divide the data into a number of clusters so that the amount of data for each cluster is less than the total data. The clustering approach does not reduce the total data, because if all the data from each cluster are added up, it will remain the same as the original data. The use of clustering, in addition to data clustering, can also be developed to become the basis for data reduction so that the amount of data is reduced. The development of data reduction, using this approach has not yet been fully explored. In the development of the IDS system, the data handled has the characteristics of a lot of noise or data outliers [13], and also has high data dimensions [14]. This condition is very appropriate when using the density-based spatial clustering algorithm of applications with the noise (DBSCAN) clustering algorithm. This is because the DBSCAN algorithm has advantages over other clustering algorithms. These advantages include the ability to detect outliers/noise. This is because the density-based concept used, namely, objects that do not

have proximity to other objects, will be recognized as outliers [15–17].

In this study, we developed a clustering model for data reduction by modifying the DBSCAN clustering algorithm (m-DBSCAN). Modifications were made so that DBSCAN carried out the clustering process, with the aim of clustering to reduce the amount of data. The use of a modification of the density-based clustering model is expected to lose information, and the differences in data homogeneity can be overcome. Modifications are made to reduce data that have a distance less than a predetermined limit (*MinNeopleshood*) to the formed clusters. A small distance value indicates that the difference in features is relatively small; therefore, if it is reduced, it does not lose information, because it is represented by other data in the cluster when it is used for training in the classification model. Reduced data are then classified using the classification and regression tree (CART) algorithm to determine the presence or absence of anomalies.

The remainder of this paper is organized into several sections. Section 2 describes the literature review, and Section 3 describes the proposed methodology. Section 4 contains the results and discussion, and the last section presents the conclusions and developments that can be continued in future work.

2. Literature Review

The dimensional reduction method is divided into two categories: reducing the amount of data and reducing the number of attributes [11]. The method of reducing the number of attributes is divided into transformation-based reduction and selection-based reduction [18]. For example, transformation-based reduction is principal component analysis (PCA), while selection-based reduction is further divided into three types: filter, wrapper, and embedded approaches. The embedded approach is a simpler method because the reduction process is carried out simultaneously with the classification process; for example, the algorithm is ID3, CART, and C4.5. The embedded method approach is more practical and it is easier to understand the resulting rules. A number of studies have shown that CART is able to provide better performance in general [19–21]. This is supported by a study conducted by Thapa et al. [22]. In this study, we developed an IDS model that combines machine learning with deep learning. The test results show that the CART performance is still better than CNN + Embedding, both in terms of accuracy and computation time. The IDS CART system also performs well in detecting normal packet types,

prob, and U2R [23]. The suitability of the CART algorithm for IDS cases was also demonstrated in a study conducted by Radoglou-Grammatikis and Sarigiannidis [24].

Algorithms in the decision tree learning group, including CART, are strongly influenced by a large amount of data. This is because each data will be partitioned and formed as a decision tree; the more data used, the higher the level of accuracy, but this also results in overlapping for the same class and criteria [25]. This will result in decision-making time and the amount of memory used. Several studies have been conducted to address these problems, including using data reduction techniques in the data preprocessing process. To perform data reduction, there are three approach strategies [26]: dimensionality reduction, numerosity reduction, and data compression. Sampling is a non-parametric method of numerosity reduction that reduces data by representing large datasets as smaller random data subsets. However, the sampling technique still has several problems in representing the data, namely, sampling can exclude some data that may not be homogeneous with the data taken [27]. This affects the level of accuracy in the classification results. The numerosity reduction approach can also be performed using clustering. This clustering approach focuses more on grouping data that have similarities [5, 6].

Data reduction models, using clustering algorithms, have been developed, see Moitra et al. [28]. In this study, data reduction was used to solve the complexity problem of persistent homology in representing topological features. K-means++ was used to form a nanocluster for each cluster formed. Subsequent research was conducted by Alweshah et al. [29]. In this research, data reduction is used to solve the problem of the amount of transaction data that affects the computation process. The proposed DRB (dataset based on clustering) method can reduce transaction data based on the cluster number of transactions that occur so that the data imbalance can be resolved. Subsequent research was conducted by Shen et al. [30], but for the case of data reduction to overcome redundant data in an image. The method used is a modification of the SVM by implementing the Fisher discriminant ratio method and the K-means method.

Le-Khac et al. [31], developed a data reduction model by combining the shared nearest neighbor similarity (SNN) algorithm and DBSCAN. The weakness in this model is that there are two processes, namely the SNN and DBSCAN processes, which, when there is a large amount of data, will affect the speed of the computation process. The development of dimensional reduction was also conducted by Wang et al. [32]. The

development carried out combines a reduction in the amount of data with random sample and attribute reduction, using PCA for further clustering using the c-means clustering algorithm. In this study, two reductions were carried out: the attributes and amount of data. The proposed reduction model combines random sampling (RS) with c-means clustering. Referring to the clustering algorithm used, the study by Le-Khac et al. [31] was better than that of Wang et al. [32]. This is due to the ability of the DBSCAN clustering algorithm to provide better performance than the c-mean, especially for noisy data [33, 34].

The current development of a clustering-based reduction model is limited to clustering data. The next step to reduce the amount of data needs to be combined with other methods, such as those conducted by Le-Khac et al. [31] and Wang et al. [32]. Merging the clustering algorithm with several other algorithms will have an impact on the speed of computation, given the large amount of data processed. This condition encourages the development of a clustering algorithm, which in the clustering process also reduces data. The data reduction process must also pay attention to performance when applied in cases of calcification, such as in an intrusion detection system.

3. Methods

3.1 Material

This research was conducted based on intrusion detection evaluation data, which is a simulation result of the US Air Force LAN military network environment. The dataset consisted of 25,192 rows and 42 data columns. The data source can be accessed online at <https://www.kaggle.com/sampadab17/network-intrusion-detection>. Furthermore, the data were divided into two categories, training and testing data. The 42 variable features used consisted of 37 quantitative features, 4 qualitative features, and 1 feature for normal and anomaly grouping. The second dataset uses the KDDCup99 dataset, which can be accessed online at <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>. The data used consisted of 39 features and 1 feature as a class whose value was normal or anomalous, with a total data of 41,166.

3.2 The Proposed Method

This study used a number of stages, as shown in Figure 1(a). Figure 1(a) shows that there are preprocessing stages, data reduction using a modified DBSCAN clustering algorithm, data sharing for training and testing, and classification using the CART algorithm; finally, we evaluate the performance of the

Figure 1. Research method.

IDS system. The preprocessing stage is divided into three processes: data cleaning, normalization, and data separation. Data normalization was performed using the z-score method [35]. The data reduction stage was performed using a modified DBSCAN algorithm. This stage is a part of the dimensional reduction in terms of the amount of data, as shown in Figure 1(b). Data reduction was developed based on the DBSCAN clustering algorithm. The DBSCAN algorithm itself is an algorithm for clustering data into a number of clusters without reducing data [14, 36]. Therefore, DBSCAN can also be used to reduce the amount of data; it is necessary to modify the DBSCAN algorithm, so that in addition to data clustering, data reduction is also performed.

Data reduction using DBSCAN was divided into three stages. The first stage separates and classifies the data based on the class. This process is performed because the method used to reduce the data is an algorithm with an unsupervised learning

model. The results of the clusters will group independently, and there is a possibility that the data will form clusters that do not match the labels contained in the data. This results in an inconsistent comparison of data ratios. The second stage reduces the data against each previously separated data, and the data reduction process is carried out using Algorithm 1. The DBSCAN algorithm is modified by adding a *MinNeighborhood* variable, which is used to determine the distance of the density to the cluster center point, which will then be marked for deletion. In the third stage, after each data class is reduced, there are no inconsistencies in the ratio of the data, the data is combined again. The distance between two points in the m-DBSCAN algorithm reduction process was measured using Eq. (1).

$$d_{ij} = \sqrt{\sum_{v=1}^n (x_{vi} - x_{vj})^2}, \quad (1)$$

where d_{ij} is the distance between point i and point j , x_{vi} coordinates of point i for the dimension v , x_{vj} coordinates of point j for dimension v , and d_{ij} is the vector of point dimension [37].

The next stage is data sharing for training and testing at a later stage. Data sharing with a composition of 70% for training and 30% for testing. The next stage is classification using the CART algorithm. This stage is divided into two parts: the training and testing processes. In the CART algorithm training section, apart from training, it is also an attribute reduction process, as shown in Figure 1(b). This is because the CART algorithm is included in the dimensional reduction with an embedded approach. The CART algorithm is a classification algorithm that uses a decision-tree classifier approach. CART aims to obtain an accurate dataset as a result of the classification. The decision tree model, produced by the CART algorithm, depends on the scale value of the dependent variable if the scale value of the response variable is continuous, the resulting decision tree model is a regression tree, and if the scale value of the response variable is categorical, the resulting tree is a classification tree [38].

The CART algorithm is a decision tree algorithm model in which there are root nodes, internal nodes, and terminal nodes. The tree depth was calculated from the main node. Constructing the CART algorithm with the classifier model is described as follows [39]:

Algorithm 1. m-DBSCAN(D, Eps, MinPts, MinNeighbor)

```

1: C = 0
2: for each unvisited point P in dataset D do
3:   mark P as visited
4:   N = getNeighbors (P, Eps)
5:   X = getNeighbors (P, MinNeighbor)
6:   if sizeof(N) < MinPts then
7:     mark P as NOISE
8:   else
9:     C = next cluster
10:  ExpandCluster(P, N, C, Eps, MinPts, MinNeighbor, X)

```

Algorithm 2. ExpandCluster(P, N, C, Eps, MinPts, MinNeighbor, X)

```

1: add P to cluster C
2: for each point P' in N do
3:   if P' is not visited then
4:     mark P' as visited
5:     N' = getNeighbors(P', Eps)
6:     deleteNeighbor = getNeighbors(P', MinNeighbor)
7:     if sizeof(N') >= MinPts then
8:       N = N joined with N'
9:       X = X joined with deleteNeighbor
10:  if P' is not yet member of any cluster then
11:    add P' to cluster C

```

3.2.1 Form a classification tree

The sample data for heterogeneous learning were used as the basis for the formation of a classification tree. At this stage, the sorting is selected from the learning data by following the sorting rules and the goodness-of-split criteria, which results in the highest reduction in heterogeneity levels. Determining the suitability of the goodness-of-split $\varnothing(s, t)$ of the candidate branch s at decision dot t is formulated as Eqs. (2)-(4).

$$N\varnothing(s, t) = 2P_L P_R Q(s | t), \quad (2)$$

$$\varnothing(s, t) = 2P_L P_R Q(s | t), \quad (3)$$

$$Q(s | t) = \sum_{j=1}^{\text{The number of categories}} |p(j | t_{left}) - p(j | t_{right})|, \quad (4)$$

where t_{left} is the left branch candidate from the decision dot t , t_{right} the right branch candidate from the decision dot t , $p(j | t_{left})$ probability j on the right branch node, and $p(j | t_{right})$ probability j at the left branch node. The sorter that results in higher $Q(s | t)$ values is the better sort and is chosen to be the node splitter/branch. A node t will become a terminal node

if there is no significant decrease in heterogeneity/impurity or a minimum limit of n , such as only one observation per child node. The minimum number of limits in an end terminal is generally 5. When terminal nodes are found, the decision tree formation is terminated.

3.2.2 Pruning the classification tree and determining the optimal classification tree

The main purpose of building a classification tree is to create a classifier model in the form of a decision tree that has the smallest misclassification error value, or close to 0. The largest classification tree must give the smallest misclassification error value, so it will always tend to choose this tree as the classifier model. However, this decision tree is highly complex in describing the data structure. One way to determine the optimum classification tree, without reducing the accuracy of this classifier model, is by trimming the classification tree. The level of importance of the tree of interest is measured based on the minimum cost complexity formulated in Eq. (5).

$$D_\alpha(T_k) = D(T_k) + \alpha |\check{T}_k|. \quad (5)$$

Dimana,

$D_\alpha(T_k)$ = devian from part tree T_k

$\check{T}_k$ = set of terminal nodes at T_k

$|\check{T}_k|$ = number of terminal nodes in $\check{T}_k$

α = cost-complexity parameter

The IDS system model developed with the DBSCAN modification, its performance was measured by referring to the confusion matrix, as shown in Table 1. The performance parameters used included accuracy, precision, sensitivity, and specificity. Referring to Table 1, the performance parameters can be calculated using Eqs. (6)-(9).

$$Sensitivity = TP_{rate} = \frac{TP}{TP + FN}, \quad (6)$$

$$Specificity = TN_{rate} = \frac{TN}{TN + FP}, \quad (7)$$

Table 1. Confusion matrixs

		Predictive class	
		Positive	Negative
Actual class	Positive	TP	FN
	Negative	FP	TN

$$Precision = \frac{TP}{TP + FP}, \quad (8)$$

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN}. \quad (9)$$

4. Result and Discussion

4.1 Results of Data Reduction with DBSCAN Modification

Data reduction is performed by deleting some data that have a certain density level that has been determined according to the data distribution. The density level is explicitly defined as the *MinNeighborhood* value. The data reduction process begins by making two experimental scenarios, namely, by reducing the data without considering labels and considering data labels. This scenario is performed to determine the effect of

data reduction by considering the data label compared without considering the data label. Reduction by considering data labels is performed by grouping the data based on labels. The next step is to calculate the distance matrix for each data to be grouped to determine the input parameters of *Eps*, *MinPts*, and *MinNeighborhood*. The last step is to reduce the data based on the cluster formed, and data reduction is performed using the density approach. In this study, the reduction scenario was carried out from data reductions of 5%, 20%, 30%, 49%, and 60%, and finally at 80%. The results of data reduction without or considering labels for the Kaggle dataset are shown in Tables 2 and 3.

Table 2 shows the results of data reduction without considering the data labels using the Kaggle dataset. In the data reduction process, the parameters of the m-DBSCAN were determined empirically. If we observe that the *MinPts* param-

Table 2. Results of data reduction without considering the data label (Kaggle)

Parameters of DBSCAN	Reduction data (%)	Number of data
Not using DBSCAN	0	25,192
MinPts=170; Eps=0.1; MinNeighborhood=0.05	5	23,932
MinPts=40; Eps=0.5; MinNeighborhood=0.05	20	20,154
MinPts=40; Eps=0.8; MinNeighborhood=0.5	30	17,634
MinPts=100; Eps=1.0; MinNeighborhood=0.8	49	12,848
MinPts=115; Eps=1.2; MinNeighborhood=0.8	60	10,080
MinPts=125; Eps=1.5; MinNeighborhood=1	80	5,040

Table 3. Results of data reduction by considering labels (Kaggle)

Reduction data (%)	Normal		Anomaly		Total number of data
	Parameter	N	Parameter	N	
0	-	13,449	-	11,743	25,192
5	MinPts=200; Eps=0.05; MinNeighborhood=0.05	12,818	MinPts=120; Eps=0.1; MinNeighborhood=0.05	11,183	24,001
20	MinPts=30; Eps=0.1; MinNeighborhood=0.1	10,812	MinPts=70; Eps=0.1; MinNeighborhood=0.05	9,431	20,243
30	MinPts=10; Eps=0.1; MinNeighborhood=0.05	9,424	MinPts=50; Eps=0.1; MinNeighborhood=0.05	8,355	17,779
49	MinPts=180; Eps=1.5; MinNeighborhood=0.1	6,663	MinPts=10; Eps=0.1; MinNeighborhood=0.05	6,005	12,668
60	MinPts=200; Eps=1.5; MinNeighborhood=0.1	5,845	MinPts=8; Eps=0.08; MinNeighborhood=0.05	4,075	9,920
80	MinPts=230; Eps=1.8; MinNeighborhood=0.2	2,851	MinPts=5; Eps=0.08; MinNeighborhood=0.05	2,161	5,012

Table 4. Results of data reduction without considering the data label (KDDCup99)

Parameters of DBSCAN	Reduction data (%)	Number of data
Not using DBSCAN	0	25,192
MinPts=10; Eps=0.1; MinNeighborhood=0.05	5	23,932
MinPts=10; Eps=0.1; MinNeighborhood=0.3	20	20,154
MinPts=10; Eps=0.25; MinNeighborhood=0.28	30	17,634
MinPts=40; Eps=0.6; MinNeighborhood=0.6	49	12,848
MinPts=115; Eps=1.2; MinNeighborhood=0.8	60	10,080
MinPts=125; Eps=1.5; MinNeighborhood=1	80	5,040

eter value will be higher when needed to produce a higher percentage of data reduction. The same is true for the *Eps* and *MinNeighborhood* parameters; the higher the value, the higher the percentage of data reduction. The m-DBSCAN parameter pattern is also the same when using label separation for data labeled normal. The data labeled anomaly shows that the smaller the value of *Eps* and *MinNeighborhood*, the higher the percentage of data reduction, while the *MinPts* parameter is relatively constant.

The next test results were obtained using the KDDCup99 dataset. Testing was performed using KDDCup99 data in 50% of the total existing data. Data were collected randomly, and data reduction was carried out using m-DBSCAN. In the KDDCup99 dataset, reduction was carried out without considering the data labels. The results of the data reduction using m-DBSCAN are shown in Table 4. The pattern of parameter values in the m-DBSCAN algorithm is the same as when using the Kaggle dataset; that is, if the *MinPts*, *Eps*, and *MinNeighborhood* values are greater, then the percentage of data reduction is also higher.

4.2 Classification Results using the CART Algorithm

The stage after data reduction is the learning and testing process of the CART algorithm, which is then followed by an evaluation of the resulting performance. The main purpose of the performance evaluation is to determine the suitability of the m-DBSCAN, combined with the CART algorithm, for IDS cases. The measurement of system performance is presented in Table 1, with the parameters measured as sensitivity, specificity, precision, and accuracy, with parameter calculation referring to Eqs. (5)-(8). The evaluation of the results of data classification, after data reduction, without considering data labels is shown in Figure 2. Figure 2 shows that the percentage of data reduction affects the IDS performance. Referring to all the performance

Figure 2. Effect of data reduction on IDS performance, not considering data labels.

Figure 3. Comparison of time computation with IDS performance, without considering labels.

parameters used, when the percentage data reduction increased, the performance decreased, but was relatively small.

The next performance evaluation of the proposed IDS system is time computation. Time computation is performed to measure the effect of percentage data reduction on the time computation. The results of the evaluation of the time computation performance parameters, for classification in the IDS system, are shown in Figure 3. Figure 3 shows the performance of the first scenario, namely, data reduction, without considering data labels. This experiment was conducted to test whether changes in the balance of the data affected the performance of the classification results. Figure 3 shows that the greater

Figure 4. Effect of data reduction on IDS performance parameters (class separation).

the percentage data reduction, the lower the computation time. This lower computation time also affected the decrease in the performance parameters of sensitivity, specificity, accuracy, and precision. From the perspective of performance, the decline is relatively small, but the decrease in computation time is very significant. For an 80% data reduction there was a decrease in computation time from 91.9 ms to 31.1 ms. The performance test was carried out using Google collaboration, with cloud specifications, using an Intel Xeon CPU @ 2.30GHz, Core i7, 25.51 RAM, and 107.77 disk.

The next test was conducted considering the labels of the data used. The reduced data, taking data labels into account, are shown in Table 3. Referring to Table 3, the classification results using CART can be shown in Figure 4. Figure 4 shows that increasing the percentage of data reduction affects the performance of the IDS system. The decline in IDS performance was not significant, especially for a relatively small specificity parameter. The sensitivity parameter is the most affected, but in general, the accuracy of the IDS performance is still relatively good with a data reduction rate of up to 80%. The pattern of decline in performance occurs linearly with the increase in amount of data reduction, but there is a different pattern when the data reduction reaches 60%. This difference is one of the effects of the empirical determination of the m-DBSCAN algorithm parameters. The selection of parameters that are not optimal reduces data that provides important information in the CART algorithm learning process [38, 39]. In addition, it is possible that when determining the m-DBSCAN parameter, data reduction of 30% and 49% is less than optimal. This results in performance that is lower than for the case of 60% data reduction.

The next test involved time computation, considering data labels. The test results are presented in Figure 5. The use of data reduction with m-DBSCAN can significantly reduce the

Figure 5. Comparison of time computation with IDS performance (class separation).

computation time while maintaining the performance of the IDS system. For example, the specificity parameter of 99.4% using 80% data reduction can maintain performance at 94.4%, with a time reduction from 91.9 ms to 31.1 ms. This also occurred for the accuracy parameter from 99.4% to 92.3%. This shows that the data reduction is up to 80%, and the IDS performance can still be maintained, indicated by the accuracy performance parameter remaining above 90%.

Testing the proposed system model uses two scenarios: testing without considering labels and scenarios considering data labels in the data reduction process. The test results of the two scenarios are shown in Figures 2-5, illustrating that the results of the two scenarios provide similar levels of performance. This condition is reinforced by the results of statistical tests, using the t-test method with a confidence level of 95%, indicating that the p -value > 0.05 , which means that there is no significant difference. This condition indicates that data reduction, carried out either by considering the data label or not considering the data label, has no effect on system performance, but in order to anticipate the occurrence of imbalanced data generated in the reduction process, the data label should be considered.

The next test uses the KDDCup99 dataset, using an approach that does not consider data labels. The results of classification testing with the CART algorithm using the KDDCup99 dataset are shown in Figure 6. The test results show that when the percentage of data reduction increases up to 80%, the proposed IDS system model has a relatively large decrease in performance at reductions of 20% and 30%, while the others are relatively the same without data reduction. This difference may be the factor determining the parameters of the M-DBSCAN algorithm, which is less than optimal, which causes the selected data to be reduced incorrectly. In the Kaggle dataset, there is a decrease in performance, although not significant. However, when using KDDCup99 the performance is the same, even though there is a small decrease. This is because the amount of data resulting

Figure 6. Effect of data reduction on IDS performance using the KDDCup99 dataset.

Figure 7. Comparison of time computation with IDS performance (KDDCup99).

from the reduction in the KDDCup99 dataset is greater than the amount of Kaggle data. In classification algorithm training, the greater the amount of data generated, the better the results [10]. The time computation performance parameter shows that the higher the percentage of data reduction, the lower it is, as shown in Figure 7. This condition indicates that the decrease in time computation is significant, but the resulting decrease in performance is relatively small. This shows that IDS using m-DBSCAN can reduce data while still providing classification performance with the CART algorithm in detecting data anomalies.

4.3 Discussion

Classification testing using three scenarios was carried out for model evaluation. An evaluation model is needed to test how well the classifier model is formed to predict a value. In the first scenario, data without reduction or separation of data were used. Based on Figure 4, it is known that the accuracy of this model is very good (99.4%), meaning that this classification model can correctly predict network anomalies in 99 out of 100 attacks. This performance requires a computation time of 91.8 ms. In the second scenario, data reduction was performed using the m-

DBSCAN algorithm without considering the data label. The use of m-DBSCAN enabled an effective reduction in data reduction of up to 80%, maintaining an accuracy of 92.0%. In addition, the computation time can also be reduced from 91.8 ms to 31.1 ms. The 31.1 milliseconds computation time is for processing 5,012 data or deleting 20,180 data. Another performance parameter of scenario two is the specificity that can be maintained from 99.4% to 95.1%, with a data reduction of 80%. This shows that m-DBSCAN can effectively reduce data by up to 80%. The third scenario is data reduction by considering labels, and the results show the same levels of performance as cases without considering data labels. This shows that data reduction can be done without considering labels because the results of reduction with m-DBSCAN are able to maintain the amount of data for each label so that it does not cause imbalanced data. M-DBSCAN is able to overcome data ratio inconsistencies, owing to the ability of the DBSCAN clustering algorithm to cluster data with a high level of compliance with data labels, so that the reduction process will not significantly affect the composition of the amount of data for each label.

In addition to the accuracy, the performance parameters measured in this study were sensitivity, specificity, and precision. The resulting sensitivity parameter shows a decrease for each increase in the percentage of reduction, but the decrease in the performance parameter is, on average, less than 7.7%. This means that the ability to detect an attack by the IDS system is also translated as a security attack; its performance decreases with a greater percentage of data reduction, but the performance only drops by an average of less than 7.7%. The highest decrease occurred at 80% data reduction. The IDS model is much more able to detect if the packet is not a security attack, as the IDS system does not translate into a security attack. This ability is indicated by the specificity performance parameter, where the average value of the decrease is less than 3.2%. The next parameter is precision, which measures the ability of the IDS system to transmit an attack, and it is a security attack. This parameter decreased by an average of less than 5%. Referring to these parameters, the use of the m-DBSCAN data reduction method in IDS shows a high reduction ability with a relatively small decrease in performance, especially when the IDS system is translating a packet that is not a security attack, and the translated IDS system is not an attack.

The clustering method has been widely used in several previous studies. Various approaches have been used, including data splitting. Data splitting is used in the development of an IDS model with a large amount of training data. In this model

Table 5. Comparison of accuracy with various data reduction methods

Reduction data (%)	Reduction Method					
	Kaggle			KDDCup99		
	m-DBSCAN	Sampling	Slice the data	m-DBSCAN	Sampling	Slice the data
0	0.994	0.994	0.994	0.944	0.944	0.944
5	0.958	0.943	0.943	0.950	0.977	0.950
20	0.955	0.956	0.941	0.944	0.960	0.950
30	0.947	0.955	0.936	0.924	0.970	0.944
49	0.936	0.937	0.923	0.935	0.947	0.941
60	0.940	0.914	0.893	0.942	0.941	0.935
80	0.923	0.903	0.865	0.930	0.930	0.930

it appears as if data are reduced because they are divided into a number of clusters, but the total data does not change. This model is used in the research of Wang et al. [4] and Wiharto et al. [12]. Both research models do not reduce the amount of data, and thus do not reduce the computation process. This is different from the proposed model, in which the clustering process is performed to reduce the amount of data. The use of the proposed model with a data reduction percentage of 80% can provide a similar level of performance achieved by work proposed by Wang et al. [4] and Wiharto et al. [12]. The data reduction model can be performed using the random down-sampling method. The random down-sampling reduction model has an impact on the resulting data; that is, when the data selection is correct, it will produce good data for training in classification, but if it is incorrect, then it will provide training results in poor classification. This also occurs in the slice of the data method, which is also done randomly in choosing from which data to reduce. In the m-DBSCAN algorithm model, the selected data are representative of the data that are included in a cluster, thus reducing the amount of data that must be eliminated.

For testing using the Kaggle dataset, data reduction using the m-DBSCAN algorithm has better capabilities than some data reduction methods, as shown in Table 5. Table 5 shows that the m-DBSCAN method has better performance for high data reduction percentages, but when the percentage of data reduction decreases, the resulting performance is lower than that of the sampling method. The results of statistical tests, using the t-test method with a confidence level of 95%, resulted in a p -value > 0.05 . These values show that the m-DBSCAN did not show a significant difference in its accuracy performance parameters, but the sampling method has problems in terms of

sampling and can exclude some data that may not be homogeneous with the data taken [27]. When compared with the slice data method, there was a significant difference. The p -value from the statistical t-test method is as much as <0.05 . The two methods, sampling and slicing of the data, are not effective when compared to m-DBSCAN. The reduction model using m-DBSCAN was also tested using the KDDCup99 dataset, and the resulting performance was not significantly different from the methods of sampling and slicing the data.

The next clustering-based reduction model is reduced through homogeneous clusters (RHC). RHC is based on the concept of homogeneity, but uses a k-means clustering algorithm [42–44]. The algorithm performs clustering for non-homogeneous data such that all the data becomes homogeneous. The weakness of this algorithm is that the use of k-means does not work well with outliers and noise datasets. In the case of IDS, it contains many outliers and noise data; therefore, algorithm selection for clustering is very important. DBSCAN has a good sensitivity capability compared to k-means [33]. Referring to research conducted by Ougiaroglou et al. [44], data reduction using the RHC algorithm and neural network classification algorithms, support vector machine, and k-NN for the percentage of data reduction of approximately 80%, the resulting accuracy was still below 90%. The RHC algorithm has a weakness in that it does not provide good performance on noise data. According to this weakness, it was developed as an extended RHC (eRHC) [45]. At eRHC when clustered with one data point, it was considered as noise and discarded. In the study of Ougiaroglou et al. [44], it was shown that data reduction with eRHC can achieve performance with an accuracy of less than 90% at a percentage of 90% data reduction, when combined with the SVM classification algorithm. Referring to the RHC and eRHC

Table 6. Comparison with previous research

Study	Method	Data reduction approach	Dataset	Sensitivity	Specificity	Accuracy
Bhattacharya et al. [46]	SVM	PCA	Kaggle	79.3	98.1	95.2
	SVM	PCA+Firefly	Kaggle	84.4	99.8	97.5
	NB	PCA	Kaggle	68.5	94.1	75.3
	NB	PCA+Firefly	Kaggle	76.8	97.2	84.2
Sarker et al. [48]	NB	-	Kaggle	90.0	-	90.0
	IntruDTree	Embed	Kaggle	98.0	-	98.0
Wang et al. [4]	BPNN+Fuzzy aggregation	Fuzzy clustering	KDDCupp99	-	-	96.6
Khare et al. [47]	DNN	SMO	KDDCupp99	92.8	93.0	92.8
	DNN	PCA	KDDCupp99	89.8	88.5	89.8
	DNN	-	KDDCupp99	90.9	88.2	90.9
Proposed	CART	m-DBSCAN	Kaggle	89.1	94.4	92.3
	CART	m-DBSCAN	KDDCupp99	93.2	94.3	93.7

algorithms, m-DBSCAN is still, overall, better than the RHC and eRHC algorithms. Related to the inconsistency ratio of the reduction result data, m-DBSCAN is better because the clustering process is carried out for all data. The basic ability of the DBSCAN algorithm can cluster data into clusters that match the data labels; therefore, the reduction process does not significantly affect changes in the composition of the data for each label. This is different from the RHC and eRHC algorithms, where clustering is applied only to homogeneous data so that it can cause inconsistency in data ratios.

The system model with the combination of m-DBSCAN and the CART algorithm in the IDS system case provides relatively good performance compared to a number of previous studies using the same dataset, namely the Kaggle dataset. A comparison with previous research using the Kaggle dataset is shown in Table 6. The ability of m-DBSCAN with CART is able to provide better than that of the naïve Bayesian (NB) method combined with PCA, even better than the combination of NB + PCA + Firefly [46]. The ability of m-DBSCAN + CART was also better than that of the model proposed by Khare et al. [47]. In this study, combining spider monkey optimization (SMO) with a deep neural network (DNN), the resulting performance is still lower than that of m-DBSCAN + CART. Compared with Wang et al. [4], it is lower, but in the case of Wang et al. [4] the approach does not reduce the amount of data, but divides the data into a fixed number of clusters, thereby increasing the complexity of the computation.

5. Conclusion

Based on the results and discussion, it can be concluded that the modified DBSCAN algorithm is sufficient and can be used to reduce large datasets. Experimental scenarios carried out with and without considering labels produce similar results, meaning that without considering data labels, DBSCAN modification can overcome the inconsistency problem of data ratio comparisons that may occur. For experimental scenarios, the parameters (*Eps*, *MinPts*, and *MinNeighborhood*) used did not have an optimal value, but depended on the distribution of data on each dataset used. Modification of DBSCAN yields good results, as shown by the test results using the Kaggle and KDDCup99 datasets, showing that the proposed system model provides performance in terms of faster computation time and is able to maintain classification accuracy above 90%.

This study empirically determines the parameters in the m-DBSCAN algorithm so that it requires repeated testing time, so it is possible that the resulting parameters are not necessarily optimal. Several heuristic algorithms can be used for further development. Heuristic algorithms that could potentially be used include genetic algorithms (GA), particle swarm optimization (PSO) algorithms, artificial bee colony (ABC) algorithms, and artificial immune system (AIS) algorithms. The use of these algorithms is expected to obtain optimal results compared with the empirical determination of DBSCAN parameters.

Conflict of Interest

No potential conflict of interest relevant to this article was reported.

Acknowledgements

We would like to thank Sebelas Maret University for providing a group research grant with contract number 260/UN27.22/HK.07.00/2021. We also express our gratitude to a number of parties who have helped to complete our research.

References

- [1] P. Angin, B. Bhargava, and R. Ranchal, "Big data analytics for cyber security," *Security and Communication Networks*, vol. 2019, article no. 4109836, 2019. <https://doi.org/10.1155/2019/4109836>
- [2] I. F. Kilincer, F. Ertam, and A. Sengur, "Machine learning methods for cyber security intrusion detection: datasets and comparative study," *Computer Networks*, vol. 188, article no. 107840, 2021. <https://doi.org/10.1016/j.comnet.2021.107840>
- [3] K. Scarfone and P. Mell, "Intrusion detection and prevention systems," in *Handbook of Information and Communication Security*. Heidelberg, Germany: Springer, 2010, pp. 177-192. https://doi.org/10.1007/978-3-642-04117-4_9
- [4] G. Wang, J. Hao, J. Ma, and L. Huang, "A new approach to intrusion detection using Artificial Neural Networks and fuzzy clustering," *Expert Systems with Applications*, vol. 37, no. 9, pp. 6225-6232, 2010. <https://doi.org/10.1016/j.eswa.2010.02.102>
- [5] A. Halimaa and K. Sundarakantham, "Machine learning based intrusion detection system," in *Proceedings of 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, Tirunelveli, India, 2019, pp. 916-920. <https://doi.org/10.1109/ICOEI.2019.8862784>
- [6] A. Meryem and B. E. Ouahidi, "Hybrid intrusion detection system using machine learning," *Network Security*, vol. 2020, no. 5, pp. 8-19, 2020. [https://doi.org/10.1016/S1353-4858\(20\)30056-8](https://doi.org/10.1016/S1353-4858(20)30056-8)
- [7] A. Phadke, M. Kulkarni, P. Bhawalkar, and R. Bhattad, "A review of machine learning methodologies for network intrusion detection," in *Proceedings of 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)*, Erode, India, 2019, pp. 272-275. <https://doi.org/10.1109/ICCMC.2019.8819748>
- [8] M. Jun and F. Shuqian, "Research of intrusion detection system based on machine learning," in *Proceedings of 2010 2nd International Conference on Computer Engineering and Technology*, Chengdu, China, 2010, pp. 713-715. <https://doi.org/10.1109/ICCET.2010.5485703>
- [9] R. Raghavan, "Study of the relationship of training set size to error rate in yet another decision tree and random forest algorithms," Ph.D. dissertation, Texas Tech University, Lubbock, TX, 2006.
- [10] M. Sordo and Q. Zeng, "On sample size and classification accuracy: a performance comparison," in *Biological and Medical Data Analysis*. Heidelberg, Germany: Springer, 2005, pp. 193-201. https://doi.org/10.1007/11573067_20
- [11] D. Uhm, S. H. Jun, and S. J. Lee, "A classification method using data reduction," *International Journal of Fuzzy Logic and Intelligent Systems*, vol. 12, no. 1, pp. 1-5, 2012. <https://doi.org/10.5391/IJFIS.2012.12.1.1>
- [12] Wiharto, A. Aziz, and U. Permana, "Improvement of performance intrusion detection system (IDS) using artificial neural network ensemble," *Journal of Theoretical and Applied Information Technology*, vol. 80, no. 2, pp. 191-201, 2015.
- [13] V. Shakya and R. R. S. Makwana, "Feature selection based intrusion detection system using the combination of DB-SCAN, K-Mean++ and SMO algorithms," in *Proceedings of 2017 international Conference on Trends in Electronics and Informatics (ICEI)*, Tirunelveli, India, 2017, pp. 928-932. <https://doi.org/10.1109/ICOEI.2017.8300843>
- [14] N. Valarmathy and S. Krishnaveni, "A novel method to enhance the performance evaluation of DBSCAN clustering algorithm using different distinguished metrics," *Materials Today: Proceedings*, 2020. <https://doi.org/10.1016/j.matpr.2020.09.623>
- [15] G. Pu, L. Wang, J. Shen, and F. Dong, "A hybrid unsupervised clustering-based anomaly detection method," *Tsinghua Science and Technology*, vol. 26, no. 2, pp. 146-153, 2020. <https://doi.org/10.26599/TST.2019.9010051>

- [16] M. A. Khadija, S. T. Widyawan, and I. L. E. Nugroho, "Detecting network intrusion by combining DBSCAN, principle component analysis and ranker," in *Proceedings of 2019 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, Yogyakarta, Indonesia, 2019, pp. 165-170. <https://doi.org/10.1109/ISRITI48646.2019.9034622>
- [17] M. Celik, F. Dadaser-Celik, and A. S. Dokuz, "Anomaly detection in temperature data using DBSCAN algorithm," in *Proceedings of 2011 International Symposium on Innovations in Intelligent Systems and Applications*, Istanbul, Turkey, 2011, pp. 91-95. <https://doi.org/10.1109/INISTA.2011.5946052>
- [18] R. Jensen and Q. Shen, *Computational Intelligence and Feature Selection: Rough and Fuzzy Approaches*. Piscataway, NJ: IEEE Press, 2008.
- [19] A. Priyam, G. R. Abhijeeta, A. Rathee, and S. Srivastava, "Comparative analysis of decision tree classification algorithms," *International Journal of Current Engineering and Technology*, vol. 3, no. 2, pp. 334-337, 2013.
- [20] H. H. Patel and P. Prajapati, "Study and analysis of decision tree based classification algorithms," *International Journal of Computer Sciences and Engineering*, vol. 6, no. 10, pp. 74-78, 2018. <https://doi.org/10.26438/ijcse/v6i10.7478>
- [21] H. M. Sani, C. Lei, and D. Neagu, "Computational complexity analysis of decision tree algorithms," in *Artificial Intelligence XXXV*. Cham, Switzerland: Springer, 2018, pp. 191-197.
- [22] N. Thapa, Z. Liu, D. B. Kc, B. Gokaraju, and K. Roy, "Comparison of machine learning and deep learning models for network intrusion detection systems," *Future Internet*, vol. 12, no. 10, article no. 167, 2020. <https://doi.org/10.3390/fi12100167>
- [23] S. Chebrolu, A. Abraham, and J. P. Thomas, "Feature deduction and ensemble design of intrusion detection systems," *Computers & Security*, vol. 24, no. 4, pp. 295-307, 2005. <https://doi.org/10.1016/j.cose.2004.09.008>
- [24] P. I. Radoglou-Grammatikis and P. G. Sarigiannidis, "An anomaly-based intrusion detection system for the smart grid based on cart decision tree," in *Proceedings of 2018 Global Information Infrastructure and Networking Symposium (GIIS)*, Thessaloniki, Greece, 2018, pp. 1-5. <https://doi.org/10.1109/GIIS.2018.8635743>
- [25] M. Mohanapriya and J. Lekha, "Comparative study between decision tree and KNN of data mining classification technique," *Journal of Physics: Conference Series*, vol. 1142, no. 1, article no. 012011, 2018. <https://doi.org/10.1088/1742-6596/1142/1/012011>
- [26] J. Han, M. Kamber, and J. Pei, "Cluster analysis: basic concepts and methods," in *Data Mining: Concepts and Techniques*, 3rd ed. Amsterdam, The Netherlands: Morgan Kaufmann, 2012, 443-495.
- [27] M. Vlachos, "Dimensionality reduction - sampling," in *Encyclopedia of Machine Learning and Data Mining*. Boston, MA: Springer, 2017, pp. 354-361.
- [28] A. Moitra, N. O. Malott, and P. A. Wilsey, "Cluster-based data reduction for persistent homology," in *Proceedings of 2018 IEEE International Conference on Big Data (Big Data)*, Seattle, WA, 2018, pp. 327-334. <https://doi.org/10.1109/BigData.2018.8622440>
- [29] M. Alweshah, W. A. AlZoubi, and A. Alarabeyyat, "Cluster based data reduction method for transaction datasets," in *Proceedings of 2015 IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE)*, Langkawi, Malaysia, 2015, pp. 78-83. <https://doi.org/10.1109/ISCAIE.2015.7298332>
- [30] X. J. Shen, L. Mu, Z. Li, H. X. Wu, J. P. Gou, and X. Chen, "Large-scale support vector machine classification with redundant data reduction," *Neurocomputing*, vol. 172, pp. 189-197, 2016. <https://doi.org/10.1016/j.neucom.2014.10.102>
- [31] N. A. Le-Khac, M. Bue, M. Whelan, and M. T. Kechadi, "A clustering-based data reduction for very large spatio-temporal datasets," in *Advanced Data Mining and Applications*. Heidelberg, Germany: Springer, 2010, pp. 43-54.
- [32] J. Wang, S. Yue, X. Yu, and Y. Wang, "An efficient data reduction method and its application to cluster analysis," *Neurocomputing*, vol. 238, pp. 234-244, 2017. <https://doi.org/10.1016/j.neucom.2017.01.059>
- [33] J. M. Dudik, A. Kurosu, J. L. Coyle, and E. Sejdic, "A comparative analysis of DBSCAN, K-means, and

- quadratic variation algorithms for automatic identification of swallows from swallowing accelerometry signals,” *Computers in Biology and Medicine*, vol. 59, pp. 10-18, 2015. <https://doi.org/10.1016/j.compbiomed.2015.01.007>
- [34] A. C. Benabdellah, A. Benghabrit, and I. Bouhaddou, “A survey of clustering algorithms for an industrial context,” *Procedia Computer Science*, vol. 148, pp. 291-302, 2019. <https://doi.org/10.1016/j.procs.2019.01.022>
- [35] D. Singh and B. Singh, “Investigating the impact of data normalization on classification performance,” *Applied Soft Computing*, vol. 97, article no. 105524, 2020. <https://doi.org/10.1016/j.asoc.2019.105524>
- [36] M. Deepa and P. Sumitra, “Detection of DOS and probe attacks based on snort with density clustering,” *Microprocessors and Microsystems*, article no. 103502, 2020. <https://doi.org/10.1016/j.micpro.2020.103502>
- [37] R. A. Johnson and D. W. Wichern, *Applied Multivariate Statistical Analysis*, 6th ed. Upper Saddle River, NJ: Prentice Hall, 2007.
- [38] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and Regression Trees*. Boca Raton, FL: CRC Press, 1984.
- [39] R. A. Berk, “Classification and regression trees (CART),” in *Statistical Learning from a Regression Perspective*. New York, NY: Springer, 2008, pp. 129-186. https://doi.org/10.1007/978-3-319-44048-4_3
- [40] W. Lai, M. Zhou, F. Hu, K. Bian, and Q. Song, “A new DBSCAN parameters determination method based on improved MVO,” *IEEE Access*, vol. 7, pp. 104085-104095, 2019. <https://doi.org/10.1109/ACCESS.2019.2931334>
- [41] A. Starczewski, P. Goetzen, and M. J. Er, “A new method for automatic determining of the DBSCAN parameters,” *Journal of Artificial Intelligence and Soft Computing Research*, vol. 10, no. 3, pp. 209-221, 2020. <https://doi.org/10.2478/jaiscr-2020-0014>
- [42] S. Ougiaroglou and G. Evangelidis, “RHC: a non-parametric cluster-based data reduction for efficient k-NN classification,” *Pattern Analysis and Applications*, vol. 19, no. 1, pp. 93-109, 2016. <https://doi.org/10.1007/s10044-014-0393-7>
- [43] S. Ougiaroglou and G. Evangelidis, “Efficient dataset size reduction by finding homogeneous clusters,” in *Proceedings of the 5th Balkan Conference in Informatics*, Novi Sad, Serbia, 2012, pp. 168-173. <https://doi.org/10.1145/2371316.2371349>
- [44] S. Ougiaroglou, K. I. Diamantaras, and G. Evangelidis, “Exploring the effect of data reduction on neural network and support vector machine classification,” *Neurocomputing*, vol. 280, pp. 101-110, 2018. <https://doi.org/10.1016/j.neucom.2017.08.076>
- [45] S. Ougiaroglou and G. Evangelidis, “Efficient editing and data abstraction by finding homogeneous clusters,” *Annals of Mathematics and Artificial Intelligence*, vol. 76, no. 3-4, pp. 327-349, 2016. <https://doi.org/10.1007/s10472-015-9472-8>
- [46] S. Bhattacharya, P. K. R. Maddikunta, R. Kaluri, S. Singh, T. R. Gadekallu, M. Alazab, and U. Tariq, “A novel PCA-firefly based XGBoost classification model for intrusion detection in networks using GPU,” *Electronics*, vol. 9, no. 2, article no. 219, 2020. <https://doi.org/10.3390/electronics9020219>
- [47] N. Khare, P. Devan, C. L. Chowdhary, S. Bhattacharya, G. Singh, S. Singh, and B. Yoon, “SMO-DNN: spider monkey optimization and deep neural network hybrid classifier model for intrusion detection,” *Electronics*, vol. 9, no. 4, article no. 692, 2020. <https://doi.org/10.3390/electronics9040692>
- [48] I. H. Sarker, Y. B. Abushark, F. Alsolami, and A. I. Khan, “IntruDTree: a machine learning based cyber security intrusion detection model,” *Symmetry*, vol. 12, no. 5, article no. 754, 2020. <https://doi.org/10.3390/sym12050754>

Wiharto is an Associate professor of Computer Science at Department of Informatics, Sebelas Maret University, Surakarta, Indonesia. He received his Ph.D. degree from Gadjah Mada University, Indonesia in 2017. He is conducting research activities in the areas of artificial intelligence, computational intelligence, expert system, network security and data mining.

telligence.

Aditya K. Wicaksana received obtained a Bachelor of Science (B.S.) from Department of Informatics, Sebelas Maret University, Surakarta, Indonesia, 2020. The area of research being carried out is the network security, data mining, artificial in-

Denis Eka Cahyani received obtained a Bachelor of Science (B.S.) from Department of Informatics, Sebelas Maret University, Indonesia, 2013 and master's degree in Computer Science (M.Cs.) from Indonesia University, Indonesia, 2015. He is presently working as a lecturer in the Department of Informatics, Sebelas Maret University, Indonesia (2021) and Department of mathematics, State University of Malang, Indonesia (2021-present). His experience and areas of interest focus on natural language processing, semantic and web information retrieval.