future internet

MDPI

Article

Classification Tendency Difference Index Model for Feature Selection and Extraction in Wireless Intrusion Detection

Chinyang Henry Tseng [1], Woei-Jiunn Tsaur [2,*] and Yueh-Mao Shen [3]

[1] Department of Computer Science and Information Engineering, National Taipei University, New Taipei City 23741, Taiwan; tsengcyt@gm.ntpu.edu.tw
[2] Computer Center, National Taipei University, New Taipei City 23741, Taiwan
[3] College of Electrical Engineering and Computer Science, National Taipei University, New Taipei City 23741, Taiwan; cna971518@gmail.com
* Correspondence: wjtsaur@mail.ntpu.edu.tw

Abstract: In detecting large-scale attacks, deep neural networks (DNNs) are an effective approach based on high-quality training data samples. Feature selection and feature extraction are the primary approaches for data quality enhancement for high-accuracy intrusion detection. However, their enhancement root causes usually present weak relationships to the differences between normal and attack behaviors in the data samples. Thus, we propose a Classification Tendency Difference Index (CTDI) model for feature selection and extraction in intrusion detection. The CTDI model consists of three indexes: Classification Tendency Frequency Difference (CTFD), Classification Tendency Membership Difference (CTMD), and Classification Tendency Distance Difference (CTDD). In the dataset, each feature has many feature values (FVs). In each FV, the normal and attack samples indicate the FV classification tendency, and CTDI shows the classification tendency differences between the normal and attack samples. CTFD is the frequency difference between the normal and attack samples. By employing fuzzy C means (FCM) to establish the normal and attack clusters, CTMD is the membership difference between the clusters, and CTDD is the distance difference between the cluster centers. CTDI calculates the index score in each FV and summarizes the scores of all FVs in the feature as the feature score for each of the three indexes. CTDI adopts an Auto Encoder for feature extraction to generate new features from the dataset and calculate the three index scores for the new features. CTDI sorts the original and new features for each of the three indexes to select the best features. The selected CTDI features indicate the best classification tendency differences between normal and attack samples. The experiment results demonstrate that the CTDI features achieve better detection accuracy as classified by DNN for the Aegean WiFi Intrusion Dataset than their related works, and the detection enhancements are based on the improved classification tendency differences in the CTDI features.

Keywords: feature selection; feature extraction; intrusion detection; Classification Tendency Difference Index; feature value; fuzzy C means; Auto Encoder; deep neural network

Citation: Tseng, C.H.; Tsaur, W.-J.; Shen, Y.-M. Classification Tendency Difference Index Model for Feature Selection and Extraction in Wireless Intrusion Detection. *Future Internet* **2024**, *16*, 25. https://doi.org/10.3390/fi16010025

Academic Editors: Weizhi Meng and Christian D. Jensen

Received: 14 December 2023
Revised: 8 January 2024
Accepted: 10 January 2024
Published: 12 January 2024

1. Introduction

As the Internet of Things influences how people live and work, wireless intrusions have also become significant threads for wireless applications [1]. Machine learning methods are effective classifiers for detecting wireless intrusions based on the training datasets [2]. As the scale of wireless intrusion increases, deep learning becomes the emerging approach for wireless intrusion detection models [3].

Machine learning models often incorporate feature selection techniques to improve the accuracy of intrusion detection. Several works [4–7] employ machine learning-based classifiers with feature selection methods for different kinds of attacks in the CIC-IDS2017 [8], NSL-KDD [9], and UNSW-NB15 [10] datasets. To improve the detection capability, a deep

neural network (DNN) combined with an Auto Encoder for feature extraction [11–14] can achieve better detection accuracy than the machine learning models.

Aegean Wi-Fi Intrusion Dataset (AWID) is a dedicated wireless attack with 154 features and consists of three major types of attacks: impersonation, injection, and flooding [15]. The ensemble of feature selection and tree-based classifiers can effectively detect the attacks in AWID [16]. The combination of DNN and Auto Encoder shows better detection accuracy than these tree-based models [17–20] in AWID. As the ensemble of DNN and Auto Encoder also integrates with feature selection and data balancing, the detection accuracy can be the best for AWID [17,18].

As these works employ feature selection and extraction to improve the detection capability, the improvements are usually evaluated by the detection accuracy, which is based on the False Positives (FP) and False Negatives (FN). Because the improvements cannot be directly explained, the root cause of the improvements is unclear. During detection, the gray areas of normal and attack behaviors in the data can cause FP and FN. Thus, the data tendency of normal and attack behaviors is the fundamental measurement for the dataset quality, and these related works cannot address this issue.

To provide a measurable feature selection model for the data tendency against normal and attack behaviors, we propose a Classification Tendency Difference Index (CTDI) model for feature selection and extraction in intrusion detections. CTDI provides three data tendency measurement methods for feature selection: Classification Tendency Frequency Difference (CTFD), Classification Tendency Membership Difference (CTMD), and Classification Tendency Distance Difference (CTDD). To identify the data tendency, CTDI targets each data value in each data feature as the essential measurement target unit. CTDI calculates the differences between normal and attack labels for the data value. CTFD directly counts the frequencies of the normal and attack labels in the feature value (FV) and calculates the absolute difference between the two frequencies. Then, CTFD computes the average count difference in the feature as a CTFD score. CTFD shows the differential frequency between normal and attack labels, and this differential frequency has a direct impact on the DNN inputs. High CTFD shows the feature has a high tendency toward the detection label, so this feature can help the classifier distinguish the attacks directly. Thus, CTFD can provide a direct data tendency measurement against the detection results. High CTFD can potentially avoid the detection of gray areas and avoid FP and FN.

CTMD and CTDD adopt fuzzy C means (FCM) to measure the data tendency. FCM provides membership values for each data value to represent the degree of relationship between normal and attack behaviors. CTMD calculates the absolute difference between the membership values of the normal and attack labels for the data value. Then, CTMD computes the average membership difference in the feature as a CTMD score. CTMD shows the differential fuzzy degree between normal and attack behaviors based on FCM clustering. High CTMD shows the feature has a high degree of data tendency toward the detection labels for high detection accuracy.

FCM also provides two cluster centers for each feature's normal and attack behaviors. CTDD calculates the distances between the data values and the two centers and the difference between the two distances. Then, CTDD computes the average distance difference in the feature as the CTDD score. CTDD shows the differential distance between the normal and attack centers of FCM clusters. High CTDD shows the feature has a high data differential distance between the two cluster centers, resulting in high detection accuracy.

CTDI selects the high data tendency features based on three data tendency measurement methods to avoid the gray areas in these features. Thus, the selected features can assist the DNN classifier in reducing potential FP and FN to improve detection accuracy. Because CTDI directly provides the relationship between input data tendency measurement and detection result labels for the DNN classifier, it provides direct evaluation assistance for FP and FN against the dataset. Therefore, the detection results in the deep learning-based intrusion detection models can be explained based on the data tendency. This can

be essential assistance in evaluating the root cause of detection errors and the potential improvement space against the dataset for the deep learning model.

CTDI adopts Auto Encoder as the feature extraction because Auto Encoder can provide additional high-quality features [11–14,17–20]. CTDI also selects the features in this input to improve the detection accuracy. Both the original and Auto-Encoder-selected data features are the inputs for the DNN classifier.

The remainder of this paper is organized as follows: Section 2 shows how the related feature selection works. Section 3 introduces the CTDI design. Section 4 illustrates the experimental results. Section 5 compares detection results with the related works in AWID. Section 6 gives the conclusion and future work.

2. Related Works

Machine learning models often incorporate data enhancement techniques, such as feature selection, Auto Encoders, and data balancing, to improve the accuracy of intrusion detection, as shown in Table 1.

Table 1. Machine learning models incorporate data enhancement techniques for intrusion detection.

Study	Classifier	Enhancement	Dataset
Alduailij et al. [4]	Random Forest	Mutual Information, Random Forest Gini impurity	CIC-IDS2017
Subbiah et al. [5]	Random Forest	Boruta Feature Selection	NSL-KDD
Alsaleh et al. [6]	XGBoost	Salp Swarm algorithm	NSL-KDD UNSW-NB15
Shah et al. [7]	Logistic Regression	Logistic Regression	NSL-KDD
Dao et al. [11]	DNN	Auto Encoder	CIC-IDS2017 UNSW-NB15
Rao et al. [12]	DNN	Auto Encoder	NSL-KDD UNSW-NB15
Bhardwaj et al. [13]	DNN	Auto Encoder	CIC-IDS2017 NSL-KDD
Yaser et al. [14]	DNN	Auto Encoder	ISCX-IDS-2012 UNSW2018
Kolias et al. [15]	J48	Manual Feature Selection	AWID
Mikhail et al. [16]	Semi-boosted Tree	Gini Impurity	AWID
Aminanto et al. [17]	DNN	Auto Encoder, three feature selection methods	AWID
Lee et al. [18]	Support Vector Machines (SVM)	Feature selection, data balancing, Auto Encoder	AWID
Parker et al. [19]	Logistic Regression	Auto Encoder	AWID
Caminero et al. [20]	Reinforcement Learning	Auto Encoder	AWID

Alduailij et al. [4] employed Random Forest as the optimal classifier and utilized a hybrid feature selection method that combines Mutual Information and Random Forest Gini impurity to enhance intrusion detection accuracy in the CIC-IDS2017 dataset [8]. Subbiah et al. [5] utilized Random Forest as the classifier and Boruta as the feature selection method to identify diverse network attacks in the NSL-KDD dataset. Alsaleh et al. [6]

employed XGBoost as the classifier and the Salp Swarm algorithm for feature selection by using both the NSL-KDD [9] and UNSW-NB15 [10] datasets. Shah et al. [7] adopted logistic regression for feature selection by using the NSL-KDD dataset.

Deep neural networks (DNNs) have become an effective classifier in intrusion detection models and have demonstrated superior accuracy compared to machine learning-based models for various attack types across different datasets. Auto Encoder has emerged as a popular data augmentation technique for enhancing DNN performance. Several studies [11–14] have successfully utilized Auto Encoder in conjunction with DNN for intrusion detection in datasets such as CIS-IDS2017, NSL-KDD, and UNSW-NB15.

In an effort to detect the three distinct AWID wireless attacks, Kolias et al. [15] employed manual feature selection and identified J48 as the most effective classifier. Mikhail et al. [16] utilized Gini impurity as the feature selection method and a semi-boosted tree as the classifier. Aminanto et al. [17] also utilized DNN with an Auto Encoder and proposed three feature selection methods with data balancing. D-FES achieved exceptionally high accuracy for impersonation attacks but did not detect flooding or injection attacks. Lee et al. [18] employed Support Vector Machines (SVM) with feature selection, Auto Encoder, and data balancing, demonstrating superior binary detection for the three AWID attacks. Parker et al. [19] utilized Logistic Regression (LR) with Auto Encoder, while Caminero et al. [20] employed reinforcement learning with Auto Encoder for binary detection in AWID.

3. Method

In this section, we first introduce the CTDI model design. Then, we illustrate the details of the three indexes as well as the usage of Auto Encoder as feature extraction in the CTDI model. Finally, an example scenario of an essential wireless feature is given to illustrate the use of CTDI.

3.1. Model Design

As Figure 1 shows, the CTDI model proposes three indexes to identify features that effectively distinguish between normal and attack network traffic. These indexes are:

Classification Tendency Frequency Difference (CTFD): Measures the difference in the frequency of occurrence of each FV between normal and attack samples.
Classification Tendency Membership Difference (CTMD): Employs FCM to cluster normal and attack samples, then calculates the membership difference between the two clusters for each FV.
Classification Tendency Distance Difference (CTDD): Calculates the distance difference between the cluster centers of normal and attack samples for each FV.

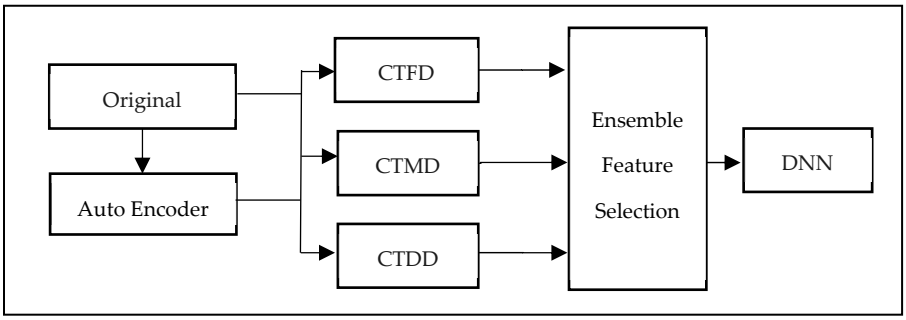

Figure 1. CTDI model design.

In addition, data preprocessing is applied to the original features. Minimax is applied to all original features to standardize their range from 0 to 1. Also, missing and unknown values are set to 0 in the original features.

For the feature extraction, the CTDI model first applies Auto Encoder to generate new extracted features from the dataset and then calculates each score of three indexes for both the original and new features. Finally, CTDI ranks the features based on their index scores and selects the top m% features for each index.

Ensemble Feature Selection (EFS): EFS is a method that combines the results of feature selection conducted using three CTDI indexes: CTFD, CTMD, and CTDD. To identify the most pertinent features, EFS first selects those that consistently appear among the top m% of features for each index. This ensures that the selected features have a high degree of representativeness. As a feature appears in two or more of the top m% features of CTDI indexes, it suggests that it has strong representativeness, and thus, this feature is chosen in CTDI. Table 2 summarizes the common abbreviations and notations in CTDI.

Table 2. Abbreviations and notations in CTDI.

CTDI	Classification Tendency Difference Index
CTFD	Classification Tendency Frequency Difference
CTMD	Classification Tendency Membership Difference
CTDD	Classification Tendency Distance Difference
EFS	Ensemble Feature Selection
DNN	Deep neural network
FCM	Fuzzy C Means
AWID	Aegean Wi-Fi Intrusion Dataset
FV_i	*i*th feature value
TF_i	Total Frequency of the repeating FV_i
C^k	The custer center for the cluster k
μ_i^k	The membership of the cluster k for FV_i
D	The size of the dataset.

3.2. CTFD

CTFD directly counts the frequency of the label k (0 is normal and 1 is an attack) in the *i*th FV, FV_i, by Equation (1), where TF_i is the total frequency of the repeating FV_i and FV_i^j is the *j*th repeating FV_i.

$$CTFD_i^k = \sum_{j=1}^{TF_i} \delta_j^k,\ \delta_j^k = \begin{cases} 1, & \text{if } FV_i^j \in \text{label k} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

Then, CTDI calculates the absolute difference between the two frequencies as Diff_CTFD and summarizes Diff_CTFD from all FVs in the feature as the CTFD feature score, $CTFD_{Score}$, by Equations (2) and (3), where D is the size of the dataset.

$$\text{Diff\_CTMD}_i = \left| CTFD_i^0 - CTFD_i^1 \right| \quad (2)$$

$$CTFD_{Score} = \sum_{i=1}^{D} \text{Diff\_CTFD}_i \quad (3)$$

3.3. CTDD and CTMD

Based on FCM, CTMD is the membership difference of the clusters, and CTDD is the distance difference of the cluster centers. As CTDI adopts FCM, it calculates the

cluster center, C^k, for the cluster k and the membership of the cluster k, μ_i^k, for FV_i by Equations (4) and (5), where D is the size of the dataset.

$$C^k = \frac{\sum_{i=1}^{D} \mu_i^k FV_i}{\sum_{i=1}^{D} \mu_i^k} \quad (4)$$

$$\mu_i^k = \frac{1}{\sum_{m=0}^{1} \left(\frac{FV_i - C^k}{FV_i - C^m} \right)^2} \quad (5)$$

While applying FCM, the fuzziness variable m is set to 2, and the number of clusters is set to 2. As C^k is randomly initialized, the first initial μ_i^k is calculated by Equation (5). The least-squares error function (6) calculates the minimum error to determine the best C^k and μ_i^k. Equations (4) and (5) keep updating C^k and μ_i^k until the minimum error is satisfied, which is set to 10^{-4}.

$$J(\mu, C) = \sum_{i=1}^{D} \sum_{k=0}^{1} \left(\mu_i^k\right)^2 \left(FV_i - C^k\right)^2 \quad (6)$$

For CTMD, CTDI calculates the absolute difference between the two FV memberships, Diff_CTMD, and summarizes Diff_CTMD from all FVs in the feature as the CTMF feature score, $CTMD_{Score}$, expressed in Equations (7) and (8).

$$Diff\_CTMD_i = \left| \mu_i^0 - \mu_i^1 \right| \quad (7)$$

$$CTMD_{Score} = \sum_{i=1}^{D} Diff\_CTMD_i \quad (8)$$

For CTDD, CTDI calculates the absolute difference of the two distances between the FV and the cluster center, Diff_CTDD, and summarizes Diff_CTDD from all FVs in the feature as the CTDD feature score, $CTDD_{Score}$, by Equations (9) and (10).

$$Diff\_CTDD_i = \left| \left| FV_i - C^0 \right| - \left| FV_i - C^1 \right| \right| \quad (9)$$

$$CTDD_{Score} = \sum_{i=1}^{D} Diff\_CTMD_i \quad (10)$$

3.4. Auto Encoder

CTDI adopts an Auto Encoder for feature extraction with similar settings to the related works [12,13,17,18], as depicted in Figure 2. The encoder transforms the original feature X into the extracted feature Y. The decoder transforms the extracted feature back to output feature Z. Y is the Auto Encoder feature extracted from X, and the size of Y is the same as X.

The encoder expressed in Equation (11) transforms X to Y with the weight W, bias b_f, and activation function f in the encoder.

$$Y = f\left(W X + b_f\right) \quad (11)$$

The decoder expressed in Equation (12) transforms Y to Z with the weight V, bias b_g, and activation function f in the decoder layer. The activation function is Relu, which is the common choice of the related works [12,13].

$$Z = f\left(V Y + b_g\right) \quad (12)$$

Auto Encoder trains the hidden representation to minimize the differences between X and Z by the loss function, which consists of Mean Square Error (MSE), L2 regularization, and Kullback-Leibler divergence and is similar to the related works [12,17,18]. MSE,

expressed in Equation (13) is the mean square error between X and Z, where D is the size of the dataset.

$$\text{MSE}\ (\text{X}, \text{Z}) \ = \ \frac{1}{\text{D}} \sum_{\text{i}\ =\ 1}^{\text{D}} (\text{X}_\text{i} - \text{Z}_\text{i})^{\ 2} \tag{13}$$

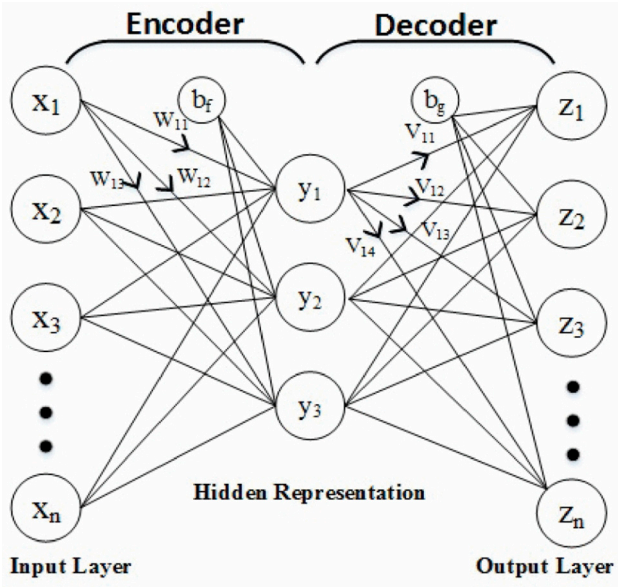

Figure 2. Auto Encoder neural network.

L2 regularization expressed in Equation (14) can avoid overfitting between the hidden layers by reducing their weights, where H is the number of neurons in the hidden layers and F is the number of features in the dataset.

$$\Omega_{\text{L2}} \ = \ \frac{1}{2} \sum_{i\ =\ 1}^{\text{H}} \sum_{j\ =\ 1}^{\text{D}} \sum_{k\ =\ 1}^{\text{F}} (\text{W}_{\text{ji}})^2 \tag{14}$$

Kullback–Leibler (KL) divergence expressed in Equation (15) calculates the sparsity regularization to prevent overfitting of the hidden neurons by reducing them. The variable ρ is the sparsity parameter, as it is near 0 to reduce neuron values. KL divergence calculates the difference between the sparsity parameter and average activation, ρ_i, to minimize their differences.

$$\begin{array}{c} \Omega_{\text{sparsity}} \ = \ \sum_{i\ =\ 1}^{\text{H}} \text{KL}\ (\rho\ ||\ \rho_i) \\ = \sum_{i\ =\ 1}^{\text{H}} \left(\rho\ log \frac{\rho}{\rho_i} + (1 - \rho) log \frac{1-\rho}{1-\rho_i} \right) \end{array} \tag{15}$$

The final loss Function (16) summarizes MSE, L2 regularization, and sparsity regularization, and the minimum error is set to 10^{-4}.

$$\textit{Error} \ = \ \text{MSE}\ (\text{X}, \text{Z}) + \ \Omega_{\text{L2}} + \Omega_{\text{sparsity}} \tag{16}$$

3.5. Example Scenario for CTDI

To illustrate the design of CTDI, an essential wireless feature, wireless LAN frame control type (wlan.fc.type), is given to illustrate the usage of CTDI.

CTFD requires calculating the frequency of the repeating feature values. For example, the *i*th feature value, FV_i, is equal to 1. This feature value repeats in 10,000 samples, of which 8000 belong to the normal ones and 2000 belong to the attack ones. Hence, TF_i is 10,000, CTFD_i^0 is 8000, CTFD_i^1 is 2000, and Diff_CTFD_i is 6000. Thus, this feature value contributes 6000 samples to $\text{CTFD}_{\text{Score}}$ for the feature wlan.fc.type. Clearly, this feature value has high differences between normal and attack samples. When other feature values also contribute high Diff_CTFD_i to $\text{CTFD}_{\text{Score}}$ becomes very high. This reveals that the feature, wlan.fc.type, presents high differences between normal and attack samples for

each of the wireless LAN type values. Thus, it this feature is valuable for detecting the attack and should be selected by CTFD.

For CTMD and CTDD, they require FCM to generate the cluster center, C^k, for the cluster k and the membership of the cluster k, μ_i^k, for FV_i. When FV_i is 1, because FV_i is close to cluster 0, μ_i^0 is high (such as 85%) and μ_i^1 is low (such as 15%). Thus, $Diff\_CTMD_i$ is also high (70%) to make $CTMD_{Score}$ of the feature high.

Besides, because FV_i is also close to C^0 and far away from C^1, $Diff\_CTDD_i$ becomes high. Therefore, $CTMD_{Score}$ of the feature also becomes high because of the many high contributions of $Diff\_CTDD_i$. As a result, this feature is selected by both CTMD and CTDD, and CTDI selects this feature based on the agreement of the three indexes.

For Auto Encoder, this feature, wlan.fc.type, is transformed into the extracted feature. Auto Encoder preserves the essential distribution of the feature values in the given feature, so the distribution difference between normal and attack samples should be highly related to the original feature. If the distribution difference in the original feature is large, the distribution difference in the extracted feature should be consistent. Thus, this extracted feature can have high scores for the three indexes and be selected by CTDI.

4. Experiment Results

4.1. Experiment Setting

The CTDI experiment program is written in Python, and the experiment software platform is Keras. The experiment is running on a desktop computer with a 2.9 GHz Intel Core i5-10400 processor and 32 GB of RAM.

AWID is the experiment dataset that has one normal class and three major attack classes: flooding, injection, and impersonation. In the adopted AWID training set, AWID-CLS-R-Trn, here are the distributions of four classes: (1) normal: 1,633,190 (91%), (2) flooding: 48,484 (3%), (3) injection: 65,379 (3%), and (4) impersonation: 48,522 (3%). To avoid training bias, the number of normal samples should be the same as the total number of attack samples in the training set. As Table 3 shows, the size of the normal class in the training set is reduced by random sampling. The AWID testing set, AWID-CLS-R-Tst, does not need to be balanced. Table 3 shows the sizes of four classes in the testing set, and their sizes remain the same. For validation purposes during the training, the training set is divided into the training and validation samples in the ratio of 8:2, and their sizes are 259,816 and 64,954, respectively.

Table 3. Distribution of normal and attack data samples.

Label	Class	Training	Test
Normal	Unbalanced	1,633,190	530,785
	Balanced	162,385	-
Attack	Impersonation	48,522	20,079
	Flooding	48,484	8097
	Injection	65,379	16,682
	Total	162,385	44,858

The experiment adopts two primary evaluation metrics: accuracy (ACC) and F1 score, which are based on the four major parameters in the confusion metrics: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). Accuracy measures the overall detection capability and is defined as follows: Accuracy = (TP + TN)/(TP + TN + FP + FN). The F1 score integrates precision and recall to evaluate the detection of attack behaviors and is defined as follows: F1 score = $(2 \times TP)/(2 \times TP + FP + FN)$. In comparison to accuracy, the F1 score directly considers both FP and FN against TP without TN. Therefore, both accuracy and F1 scores are valuable for evaluating the effectiveness of intrusion detection models.

The AWID dataset has 154 features, which are designed to detect the three wireless attacks: flooding, injection, and impersonation. These features are mainly related to Wi-Fi

major parameters for frames, radio channels, wireless LAN control, and management. From these original features, CTDI generates 154 Auto Encoder features. Then, CTDI selects the effective features from both the original and Auto Encoder data samples.

4.2. Feature Selection Results

Figure 3 shows the example in CTDD using original features as we determine m%. As m% is 10%, clearly, accuracy is the best. For the other CTDI index cases, the results are similar. Thus, m% is set to 10%, and the 16 features are selected for each index from the 154 features.

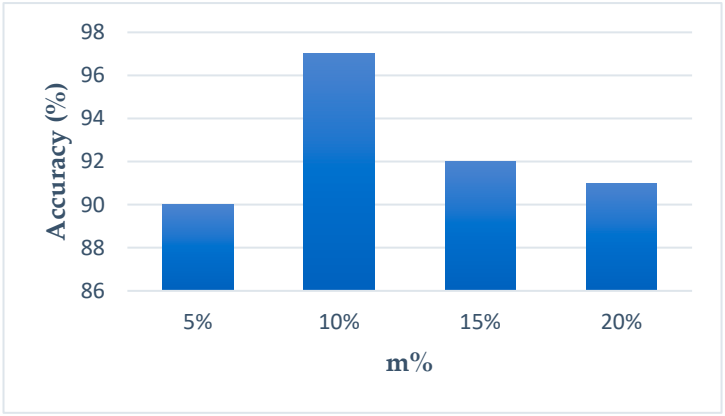

Figure 3. The accuracy for different m%.

Table 4 shows the feature selection results. For original features, the selected features are similar, so EFS can have 18 features from them. This shows the selected 18 features are representative since the three indexes reach a high consensus on these features. The Auto Encoder features are extracted from the original features, and their features are less similar than the original features. EFS selects six features from them. Overall, EFS in CTDI selects 24 features from both the original and Auto Encoder features.

Table 4. Feature selection results.

Method	Selected Features	Number of Features
	Original Feature	
CTFD	4, 7, 8, 9, 38, 47, 50, 51, 64, 67, 70, 71, 73, 140, 142, 154	16
CTMD	8, 9, 47, 50, 51, 64, 66, 67, 68, 70, 71, 73, 90, 118, 142, 154	16
CTDD	4, 7, 8, 9, 47, 50, 51, 66, 67, 68, 71, 82, 118, 140, 142, 154	16
EFS	4, 7, 8, 9, 47, 50, 51, 64, 66, 67, 68, 70, 71, 73, 118, 140, 142, 154	18
	Auto Encoder Feature	
CTFD	10, 16, 24, 27, 35, 49, 51, 55, 68, 76, 78, 82, 87, 109, 139, 143	16
CTMD	8, 13, 22, 30, 39, 42, 66, 69, 76, 78, 100, 103, 115, 118, 139, 141	16
CTDD	5, 8, 9, 31, 39, 41, 50, 54, 66, 71, 75, 80, 86, 92, 98, 102, 112, 147	16
EFS	8, 39, 66, 76, 78, 139	6
CTDI	Original: 4, 7, 8, 9, 47, 50, 51, 64, 66, 67, 68, 70, 71, 73, 118, 140, 142, 154; Auto Encoder: 8, 39, 66, 76, 78, 139	24

Table 5 shows the names of selected original and Auto Encoder features in the CTDI. The #8 feature, frame length, and #66 feature, WLAN frame control type, are selected in both original and Auto Encoder features, so they are recognized as the critical features to distinguish the attack from normal behaviors. The #66 feature, WLAN frame control type,

is clearly the key feature to identify wireless network behaviors by its definition. Some type-related features, such as #50, #51, #64, and #67, are also selected to distinguish the attack behaviors. It shows that the CTDI model can identify features with clear differences between normal and attack behaviors.

Table 5. The names of selected original and Auto Encoder features in CTDI.

Original Feature					
No.	**Name**	**No.**	**Name**	**No.**	**Name**
4	frame.time_epoch	51	radiotap.channel.type.ofdm	71	wlan.fc.pwrmgt
7	frame.time_relative	64	wlan.fc.type_subtype	73	wlan.fc.protected
8	frame.len	66	wlan.fc.type	118	wlan_mgt.tagged.all
9	frame.cap_len	67	wlan.fc.subtype	140	wlan.wep.iv
47	radiotap.datarate	68	wlan.fc.ds	142	wlan.wep.icv
50	fradiotap.channel.type.cck	70	wlan.fc.retry	154	data.len
Auto Encoder Feature					
8	frame.len	66	wlan.fc.type	78	wlan.ta
39	radiotap.flags.cfp	76	wlan.ra	139	wlan_mgt.tcprep.link_mrg

The packet length features #8 frame.len, #9 frame.cap_len, and #154 data.len are also selected to detect the attack behaviors. Other selected original features are related to notable wireless parameters, such as time (#4 and #7), frame control (#70, #71, and #73), and WEP (#140 and #142). Auto Encoder features are extracted from the original features, so their selected features show four different wireless parameters (#39, #76, #78, and #139).

4.3. Detection Results

Table 6 shows the detection results for all sets of selected features. Of the 24 selected features, CTDI achieves the best results with 99.92% accuracy and a 99.52% F1 score, which are very convincing for detecting AWID wireless attacks. For the originally selected feature sets, they also achieved very good results in accuracy and F1 score. This shows these selected features are very representative and match the observation of feature selection results. CTFD shows the best results for the original selected feature sets. For the Auto Encoder features, CTMD also shows the best detection results, so the Auto Encoder selected features are also very helpful.

Table 6. Detection results from all sets of selected features.

Method	**Accuracy (%)**	**F1 (%)**	**Precision (%)**	**Recall (%)**
CTDI	99.92%	99.52%	99.98%	99.06%
Original Feature				
CTFD	98.31%	90.08%	98.32%	83.12%
CTMD	98.07%	88.72%	97.14%	81.64%
CTDD	97.10%	83.29%	92.74%	75.58%
Auto Encoder Feature				
CTFD	94.65%	73.11%	93.35%	60.09%
CTMD	98.25%	89.52%	95.86%	83.97%
CTDD	97.43%	85.03%	93.75%	77.80%

The results show that the selected features representing the high differences between normal and attack classes are very effective in distinguishing between normal and attack behaviors in wireless networks. CTDI accumulates the best selected features from the original and Auto Encoder features, and thus it achieves the best convincing detection results.

The EFS in CTDI adopts the majority voting approach to select the features from the three indexes as two of the indexes agree with the same feature. Other related works adopting the ensemble approach usually select several machine learning methods, such as Random Forest and Mutual Information [4], Decision Tree 4.5, Support Vector Machine, and Artificial Neural Network [17]. However, they do not merge the selected features from the different feature selection methods. The results show that CTDI achieves better results than CTFD, CTMD, and CTDD, and thus CTDI shows the merged feature sets by EFS are more effective than the related works, which only use one feature selection method to generate one separated selected feature set individually.

5. Comparing Detection Results with the Related Works in AWID

Table 7 shows the detection results of the related works in AWID compared with the proposed CTDI model. CTDI applies the three data enhancement approaches: feature selection (FS), data balancing (DB), and Auto Encoder (AE), and CTDI achieves the best results for detecting all three attacks. Aminanto et al. [17] only detected impersonation attacks, so they could achieve the highest results. As we reproduce the results of its selected features against the three attacks, its results become much lower than those of CTDI. The author of AWID [15] adopted J48 as its best classifier with FS. Its results show that the baseline of AWID is lower than that of CTDI.

Table 7. Comparing detection results with the related works in AWID.

Study	Classifier	Enhancement	ACC	F1
Kolias et al. [15]	J48	FS	96.28%	68.86%
Aminanto et al. [17]	DNN	FS, DB, AE	99.97%	99.94%
D-FES Reproduced	DNN	FS, DB, AE	96.11%	78.40%
Lee et al. [18]	SVM	FS, DB, AE	98.22%	98.21%
Parker et al. [19]	LR	AE	98.04%	98.01%
Caminero et al. [20]	RL	AE	95.90%	96.29%
Mikhail et al. [16]	Semi-boosted	FS, DB	95.26%	82.09%

Lee et al. [18] adopted SVM as the classifier and applied FS, DB, and AE, and it shows the second-best results in Table 7. Parker et al. [19] adopted Logical Regression as the classifier with AE, and its results are slightly lower than Lee et al.'s method [18]. They both adopt machine learning-based classifiers with AE to achieve good detection results, but they are still worse than CTDI. The combination of reinforcement learning (RL) with AE [20] shows lower accuracy, but its F1 is relatively high. This shows RL is less effective in detecting normal behaviors in AWID. The semi-boosted tree with FS and DB [16] has lower results than those employing DNN or AE. In conclusion, the proposed CTDI shows the best detection result among the related works for the three types of attacks in AWID. This shows that the proposed CTDI, which chooses the selected features with high differences between normal and attack behaviors, is very effective for wireless attacks.

6. Conclusions

In this paper, we propose CTDI, which consists of three indexes: CTFD, CTMD, and CTDD for feature selection for original and Auto Encoder features, to detect three types of wireless attacks in AWID. After integrating the results of feature selection by three CTDI indexes, we extract the most relevant features that show high differences between normal and attack behaviors. During the CTDI process, 18 features are selected from the original features, and 6 features are selected from the AE features, for a total of 24 features. The experimental results demonstrate that the CTDI features achieve 99.92% accuracy and a 99.52% F1 score with the DNN classifier, outperforming the related works in AWID. Unlike other related works adopting the ensemble approach and only using the individual feature

selection method to select features separately, the proposed CTDI merged the features from the three indexes by EFS, the majority voting. The results show that CTDI achieves better results than CTFD, CTMD, and CTDD, which are individual feature selection indexes. Therefore, EFS enables CTDI to achieve better results than the individual feature selection method adopted by other ensemble works.

Besides, other network attack datasets, such as UNSW-NB15 and CIC-IDS2017, have similar issues with AWID. These datasets also require selecting features containing clear differences between normal and attack behaviors. Our feature work will apply CTDI to these datasets to discover effective features, as we found in AWID.

Author Contributions: Conceptualization, C.H.T.; methodology, C.H.T.; software, C.H.T.; validation, C.H.T. and W.-J.T.; formal analysis, C.H.T. and W.-J.T.; investigation, Y.-M.S.; resources, C.H.T.; data curation, C.H.T.; writing—original draft preparation, C.H.T. and Y.-M.S.; writing—review and editing, W.-J.T.; visualization, C.H.T.; supervision, W.-J.T.; project administration, W.-J.T.; funding acquisition, W.-J.T. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the National Science and Technology Council in Taiwan under contract numbers NSTC 111-2221-E-305-005-MY2 and NSTC 112-2622-E-305-004.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to location privacy.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Badii, C.; Bellini, P.; Difino, A.; Nesi, P. Smart City IoT Platform Respecting GDPR Privacy and Security Aspects. *IEEE Access* **2020**, *8*, 23601–23623. [CrossRef]
2. Al Lail, M.; Garcia, A.; Olivo, S. Machine Learning for Network Intrusion Detection—A Comparative Study. *Future Internet* **2023**, *15*, 243. [CrossRef]
3. Aldweesh, A.; Derhab, A.; Emam, A.Z. Deep learning approaches for anomaly based intrusion detection systems: A survey, taxonomy, and open issues. *Knowl.-Based Syst.* **2020**, *189*, 105124. [CrossRef]
4. Alduailij, M.; Khan, Q.W.; Tahir, M.; Sardaraz, M.; Alduailij, M.; Malik, F. Machine-Learning-Based DDoS Attack Detection Using Mutual Information and Random Forest Feature Importance Method. *Symmetry* **2022**, *14*, 1095. [CrossRef]
5. Subbiah, S.; Anbananthen, K.S.M.; Thangaraj, S.; Kannan, S.; Chelliah, D. Intrusion detection technique in wireless sensor network using grid search random forest with Boruta feature selection algorithm. *J. Commun. Netw.* **2022**, *24*, 264–273. [CrossRef]
6. Alsaleh, A.; Binsaeedan, W. The Influence of Salp Swarm Algorithm-Based Feature Selection on Network Anomaly Intrusion Detection. *IEEE Access* **2021**, *9*, 112466–112477. [CrossRef]
7. Shah, R.A.; Qian, Y.; Kumar, D.; Ali, M.; Alvi, M.B. Network Intrusion Detection through Discriminative Feature Selection by Using Sparse Logistic Regression. *Future Internet* **2017**, *9*, 81. [CrossRef]
8. Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A.A. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP), Funchal, Portugal, 22–24 January 2018.
9. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 8–10 July 2009.
10. Moustafa, N.; Slay, J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In Proceedings of the Military Communications and Information Systems Conference (MilCIS), Canberra, Australia, 10–12 November 2015.
11. Dao, T.-N.; Lee, H. Stacked Autoencoder-Based Probabilistic Feature Extraction for On-Device Network Intrusion Detection. *IEEE Internet Things J.* **2022**, *9*, 14438–14451. [CrossRef]
12. Rao, K.N.; Rao, K.V.; Prasad Reddy, P.V.G.D. A hybrid Intrusion Detection System based on Sparse autoencoder and Deep Neural Network. *Comput. Commun.* **2021**, *180*, 77–88.
13. Bhardwaj, A.; Mangat, V.; Vig, R. Hyperband Tuned Deep Neural Network with Well Posed Stacked Sparse AutoEncoder for Detection of DDoS Attacks in Cloud. *IEEE Access* **2020**, *8*, 181916–181929. [CrossRef]
14. Yaser, A.L.; Mousa, H.M.; Hussein, M. Improved DDoS Detection Utilizing Deep Neural Networks and Feedforward Neural Networks as Autoencoder. *Future Internet* **2022**, *14*, 240. [CrossRef]
15. Kolias, C.; Kambourakis, G.; Stavrou, A.; Gritzalis, S. Intrusion Detection in 802.11 Networks: Empirical Evaluation of Threats and a Public Dataset. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 184–208. [CrossRef]
16. Mikhail, J.W.; Fossaceca, J.M.; Iammartino, R. A Semi-Boosted Nested Model with Sensitivity-Based Weighted Binarization for Multi-Domain Network Intrusion Detection. *ACM Trans. Intell. Syst. Technol.* **2019**, *10*, 1–27. [CrossRef]

17. Aminanto, M.E.; Choi, R.; Tanuwidjaja, H.C.; Yoo, P.D.; Kim, K. Deep Abstraction and Weighted Feature Selection for Wi-Fi Impersonation Detection. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 621–636. [CrossRef]
18. Lee, S.J.; Yoo, P.D.; Asyhari, A.T.; Jhi, Y.; Chermak, L.; Yeun, C.Y.; Taha, K. IMPACT: Impersonation Attack Detection via Edge Computing Using Deep Autoencoder and Feature Abstraction. *IEEE Access* **2020**, *8*, 65520–65529. [CrossRef]
19. Parker, L.R.; Yoo, P.D.; Asyhari, T.A.; Chermak, L.; Jhi, Y.; Taha, K. DEMISe: Interpretable deep extraction and mutual information selection techniques for IoT intrusion detection. In Proceedings of the International Conference on Availability, Reliability and Security (ARES '19), New York, NY, USA, 26–29 August 2019.
20. Caminero, G.; Lopez-Martin, M.; Carro, B. Adversarial environment reinforcement learning algorithm for intrusion detection. *Comput. Netw.* **2019**, *159*, 96–109. [CrossRef]