Article

FCNN-SE: An Intrusion Detection Model Based on a Fusion CNN and Stacked Ensemble

Chen Chen [1,2], Yafei Song [2], Shaohua Yue [2,*], Xiaodong Xu [1], Lihua Zhou [1], Qibin Lv [3] and Lintao Yang [1]

1 Xi'an Satellite Control Center, Xi'an 710043, China
2 College of Air and Missile Defense, Air Force Engineering University, Xi'an 710051, China
3 Graduate School, Academy of Military Sciences, Beijing 100097, China
* Correspondence: shaohua_yue@163.com

Abstract: As a security defense technique to protect networks from attacks, a network intrusion detection model plays a crucial role in the security of computer systems and networks. Aiming at the shortcomings of a complex feature extraction process and insufficient information extraction of the existing intrusion detection models, an intrusion detection model named the FCNN-SE, which uses the fusion convolutional neural network (FCNN) for feature extraction and stacked ensemble (SE) for classification, is proposed in this paper. The proposed model mainly includes two parts, feature extraction and feature classification. Multi-dimensional features of traffic data are first extracted using convolutional neural networks of different dimensions and then fused into a network traffic dataset. The heterogeneous base learners are combined and used as a classifier, and the obtained network traffic dataset is fed to the classifier for final classification. The comprehensive performance of the proposed model is verified through experiments, and experimental results are evaluated using a comprehensive performance evaluation method based on the radar chart method. The comparison results on the NSL-KDD dataset show that the proposed FCNN-SE has the highest overall performance among all compared models, and a more balanced performance than the other models.

Keywords: intrusion detection; feature extraction; fusion CNN; stacked ensemble; radar chart method

check for updates

Citation: Chen, C.; Song, Y.; Yue, S.; Xu, X.; Zhou, L.; Lv, Q.; Yang, L. FCNN-SE: An Intrusion Detection Model Based on a Fusion CNN and Stacked Ensemble. *Appl. Sci.* **2022**, *12*, 8601. https://doi.org/10.3390/app12178601

Academic Editors: Tarek Gaber, Shu-Chuan Chu and Chin-Shiuh Shieh

Received: 27 July 2022
Accepted: 25 August 2022
Published: 27 August 2022

1. Introduction

With the advent of big data, network information has been facing an increasing number of security threats. Although the rapid development of communication networks makes the information exchange and data transmission between users more convenient, it brings the risk of various attacks on users' private data. It should be noted that once a network is malfunctioned by unknown attacks, malicious leakage and illegal use of important information can easily occur, resulting in enormous losses [1]. Therefore, how to predict and deal with network attacks timely and effectively has always been a research hotspot in the network security field. In the view of network security, the concept of intrusion detection was first proposed by Anderson in 1980 [2], and then many detection models represented by intrusion detection expert systems [3] were developed. These models monitor the network operation status using certain software and hardware selected according to security policies and detect as many intrusions as possible to prevent damage to the network and data. Network intrusion detection, as a proactive security protection technology that aims to intercept and respond to intrusions before the network system is compromised, has received extensive attention worldwide. Traditional intrusion detection methods include misuse detection and anomaly detection. Misuse detection finds anomalous links in a network by establishing an intrusion rule base. Although this method has a high accuracy rate, it is often ineffective for new types of intrusions and old virus variant connections [4]. Anomaly detection is used in network anomaly analysis by summarizing the characteristics of normal network connections, and this method has gained widespread attention because it has a good detection effect for new types of attacks [5]. However, due to increasingly complex network environments, the defects of both misuse- and anomaly-based intrusion detection systems, such as high resource consumption, slow detection speed, and the need for manual intervention, have been increasingly prominent. It should be noted that when anomalous access or connection events are detected and handled, many severe consequences may have already occurred. Therefore, efficient and fast intrusion detection has been an extremely challenging task [6].

Currently, many machine learning models, including discriminant analysis (DA) [7], the *K*-nearest neighbor (KNN) algorithm [8], decision tree (DT) [9], naive Bayes (NB) [10], logistic regression (LR) [11], and support vector machine (SVM) [12], have been widely used in intrusion detection. First, historical access data are extracted from a database, where each historical access data sample contains certain information about accesses, such as access duration and usage of transmission control protocol (TCP) or user datagram protocol (UDP). Then, these data are labeled to mark normal and abnormal accesses, and finally, are used as training data for a machine learning algorithm. Training of supervised machine learning algorithms results in a classification model for intrusion detection and prediction of unknown accesses. Machine learning-based intrusion detection methods can have both advantages and disadvantages depending on a selected learner (classifier) [13]. In general, complex classifiers require a relatively long time to train and are time-consuming, while simple learners, although efficient in processing, may be difficult to ensure effective detection for network attacks with a mixture of multiple features and complex and variable intrusion methods.

Some studies have aimed to improve detection performance by combining multiple learners, and intrusion detection methods based on ensemble learning have received extensive attention in recent years [14]. Ensemble learning algorithms can improve the overall algorithm generalization capability by combining multiple base learners. Theoretically, intrusion detection based on ensemble learning is much better for the identification of unknown network attacks than intrusion detection methods based on a single learner. In ensemble learning, the early boosting algorithm was first used for practical applications by Schapire in 1993 [15]. The early boosting algorithm combines multiple weak learners into one strong learner. Subsequently, Freund and Schapire proposed an improved boosting algorithm named the Adaboost (i.e., adaptive boosting) in 1995 [16]. The Adaboost algorithm is efficient and has been widely used in practice. The bagging algorithm was first proposed by Breiman in 1996 [17] to improve the accuracy and stability of the computing while avoiding overfitting by reducing the variance of results. However, both boosting and bagging algorithms are homologous assemble, namely, the base learners use models with the same structure. In multi-classification scenarios, due to the differences in design principles and actual performances of different models, using models with the same structure will inevitably result in different detection rates for different types of data. To solve the problem that a homogeneous ensemble cannot overcome the low detection rate for certain data types using the same structural model, the SE algorithm has been proposed as a heterogeneous ensemble algorithm to improve the detection rate of all data types comprehensively [18]. The SE method tends to combine the advantages of different base classifiers through a certain strategy to improve classification efficiency.

The cyberspace data stream contains a large amount of temporal, spatial, load, and statistical data [19], and it may also contain some incomplete or redundant information that may affect the data analysis process and results. Therefore, anomaly detection models based on full data features cannot capture anomalous information hidden in local data features. However, data analysis from different dimensions and perspectives can provide certain contributions and support to anomaly detection and analysis results. Commonly, the analysis implies a mutually supportive and complementary relationship between basic features of different dimensions [20]. Nonetheless, anomalies are usually manifested in multiple basic feature data, so a comprehensive feature dataset is required. Feature fusion is based on effective feature extraction of the original data, and deep learning has been successfully applied to the field of feature extraction [21]. Convolutional neural networks

(CNNs) [22,23] denote deep learning-based models with a convolutional structure, which have been mainly used in the fields of computer vision and natural language processing. CNN-based methods for feature extraction generally use a single-scale single-type convolution kernel to extract target features, which may bring the problem of incomplete and inaccurate details of the extracted features [24].

With the in-depth research on intrusion detection, an increasing number of models have been proposed to solve the intrusion detection problem; there are certain performance differences between these models. Currently, in the evaluation of the intrusion detection model performance, several evaluation metrics, including accuracy, precision, recall, specificity, and F1 score, have been mainly used to evaluate the model performance [25–27]. However, the evaluation results may not be unique due to the inconsistency in evaluation metrics used in the comparative experiments. Additionally, a single-metric evaluation cannot fully reflect the comprehensive performance of a model. Commonly used comprehensive evaluation methods can be roughly classified into expert evaluation methods, economic analysis methods, operational research methods, mathematical and statistical methods, and radar chart methods. The radar chart methods denote common graphical methods for displaying multiple variables [28], which can map a multi-dimensional space point to a two-dimensional space and can evaluate each evaluation object qualitatively. In addition, these methods can construct specific evaluation vector and function by extracting the feature vector of a radar plot and then use the evaluation function magnitude to realize a comprehensive evaluation of the selected evaluation object [29]. The main advantages of these methods are that they are intuitive, visual, and easy to operate. They denote typical graphical evaluation methods with a wide range of engineering applications.

The specific contributions of this paper are as follows:

A feature extraction method, which uses the fusion CNN (FCNN) to fuse 1DCNN and 2DCNN, which is then used to extract features of the NSL-KDD dataset to generate a network traffic dataset, is proposed. The FCNN can ensure that low- and high-dimensional feature information is extracted simultaneously from the original dataset, and the generated feature set can characterize the specific attack mode comprehensively and in detail.

An innovative classification model based on the SE, with a heterogeneous learner as a base learner and an SVM as the meta-learner, is proposed for intrusion detection. Detailed simulation experiments and analyses are conducted to verify the proposed model;

A comprehensive performance evaluation index based on the radar chart method, which can achieve a comprehensive evaluation of comprehensive performance, is designed and used as an evaluation index of the intrusion detection model.

An ablation study, which illustrates the effect of feature extraction using CNNs of different dimensions on data distribution, is conducted.

The rest of this paper is structured as follows. Section 2 discusses the related work. Section 3 describes the proposed method. Section 4 conducts simulation experiments and results analysis. Finally, Section 5 presents the research conclusions.

2. Related Work

A well-performing intrusion detection model must be able to perform self-learning, self-adaptation, and eventually alarm various connection violations at high speed and low false alarm rate and missing rate. In recent years, network intrusion detection methods based on machine learning have received extensive attention due to their strong self-adaptability performance and high intelligence [30]. Machine learning-based network intrusion detection aims to transform the network intrusion detection problem into the pattern recognition (classification) problem. Zheng et al. [31] combined the DA with an extreme learning machine to classify dimensionality-reduced data using an extreme learning machine with a single hidden layer under the premise of reducing the feature size. They simplified the classification model structure and achieved an accuracy of 92.35% on the NSL-KDD dataset. Labiod et al. [32] proposed an intrusion detection model, which combines fog computing combing variational autoencoders and multilayer perceptrons, to develop an efficient distributed lightweight intrusion detection system. This system adopts a two-layer fog architecture with anomaly detectors inside the fog nodes and attack recognition modules in the cloud and thus can accurately characterize the normal behavior inside the fog nodes and detect different types of attacks, such as DDoS attacks. Saba et al. [33] reduced the information component of CNN by sharing parameters, equivariant representations, and sparse connections, as well as links between the layers, thus extending the scalability and reducing the training time. Yu et al. [34] designed a multi-scale CNN model with multiple convolutional layers, where multi-scale convolutional layers and two convolutional layers were added and connected to the pooling layer through a multi-scale convolutional layer, and finally, to the softmax classifier through three fully connected layers. Although these approaches can provide the desired performance, it is difficult for an intrusion detection system composed of a single model to realize effective monitoring and processing, so the intrusion detection systems can scarcely meet the requirements of integrity and parallelism at the same time.

To address this problem, various ensemble learning-based models have been proposed to solve the intrusion detection problem. Wu et al. [35] used the RVMs as base learners, determined voting weights for each RVM base learner dynamically, and obtained the final ensemble model classification results using the voting mechanism. This model has certain advantages in terms of the time cost and storage space. Mokbal et al. [36] extracted an effective subset of features using an embedded feature selection method, which focuses on extracting features that can be computed rapidly and correctly using a relative importance approach. This method analytically selects the best features that can represent all attacks uniformly and comprehensively rather than selecting features for each attack separately. Finally, extreme gradient boosting was used to perform intrusion detection on the feature subset, which improved the detection accuracy. Alanazi et al. [37] used four machine learning techniques, namely, the decision tree, extra tree, random forest, and XGBoost, to select the best features independently; features that obtained high scores were added to the best feature set, and then the best feature set's features were classified by the ensemble classifier that combines multiple decision trees. This method can select the best unique features and eliminate unnecessary features, providing effective and efficient feature detection. The above-mentioned models can improve the overall detection accuracy by combining multiple homologous algorithms, but homologous model-based ensemble suffers from the problems of a model's overfitting and insufficient generalization ability.

To improve the intrusion detection accuracy further while improving the classification performance, a large number of methods for feature selection and optimization have been proposed and verified on different datasets. Prasad et al. [38] proposed a feature selection method based on multilevel correlation, which selects important features and reduces the training set size based on the multilevel correlation between the features; the verification results have indicated that this method can successfully improve detection capability. Patil et al. [39] used feature selection methods, such as relevance-based feature subset selection, chi-square attribute evaluation, gain ratio attribute evaluation, and information gain attribute evaluation, to improve the data quality. The results have indicated that by adopting a feature selection strategy, considerable acceptable attack detection accuracy can be obtained at minimal system overhead. Quincozes et al. [40] designed a feature extraction method named the GRASP-FS, which uses the F1 score as a fitness function for adapting the greedy random adaptive search process (GRASP) meta-heuristic. The experiments on the SWaT dataset showed that the GRASP-FS constructed a simplified subset of five features from 51 available features and used random trees as classifiers, achieving the F1 score and accuracy of 96.97% and a 99.65%, respectively.

All the above-mentioned feature extraction methods can reduce the feature space size of the original dataset and improve the detection performance of a classifier to a certain extent. However, high-dimensional feature information cannot be extracted from a dataset using feature extraction methods that only reduce the feature space size.

Recently, a large number of evaluation metrics based on statistical principles have been applied in the field of intrusion detection to evaluate the performances of intrusion detection systems. Prakash et al. [41] used the DT, LR, KNN, SVM, and Bi-LSTM as base learners and adopted the voting mechanism for classification. The weights of all base learners were optimized using a hybrid approach of particle swarm optimization and a modified salps swarm algorithm. The proposed ensemble classifier achieved good performances in terms of accuracy rate, attack detection rate, and false alarm rate. Babu et al. [42] proposed a bat-inspired optimization and correlation-based feature selection (BIOCFS) algorithm, which can estimate the correlation between identified features and select optimal subsets for the training and testing phases. In the BIOCFS algorithm, the base learner in the ensemble classifier uses the forest by penalizing attributes, random forest, and C4.5. The BIOCFS algorithm has an excellent performance in handling multiclassification and unbalanced datasets. On the NSL-KDD dataset, the BIOCFS algorithm can achieve a maximum classification accuracy, precision, F1 score, and attack detection ratio of 0.994, 0.993, 0.992, and 0.992, respectively, and the minimum false alarm rate of 0.008%. Niu et al. [43] proposed a multi-granularity feature generation algorithm, which converts features into discrete features with different numbers of classes, where different numbers of classes indicate different granularities. On the KDD99 dataset, the multi-granularity feature generation algorithm can achieve the detection rates of 100%, 100%, and 99.43% for two-, five-, and multi-class tasks, respectively; on the NSL-KDD dataset, this algorithm can achieve the detection rates of 100%, 100%, and 90.84% for the two-, five-, and multi-class tasks, respectively. All aforementioned intrusion detection models can achieve excellent detection results, but their comprehensive performance cannot be fully evaluated using a single evaluation metric.

In this paper, an intrusion detection model (FCNN-SE) based on the FCNN and SE, which can effectively detect attacks in networks with a high traffic data rate, is proposed. The proposed model divides network data into two parts that are input to the FCNN at the same time; one part denotes one-dimensional data that are input directly into the FCNN, and the other part is the remaining data that are first converted into a two-dimensional matrix and then input into the FCNN. The feature information obtained from the two-dimensional CNN is fused in the aggregation layer to obtain a new dataset. The set of heterogeneous base learners is combined into an ensemble classifier SE, and the new dataset is input to SE for classification to ensure accurate detection of different attack types.

3. Materials and Methods

The proposed FCNN-SE consists of two modules, a feature extraction module and a SE-based intrusion detection module. The FCNN is used as a feature extractor to construct a new dataset and design classifiers to perform intrusion detection using an ensemble learning approach based on stacking. The overview structure of the proposed FCNN-SE is shown in Figure 1.

3.1. FCNN

In the process of FCNN forward propagation, first, convolution kernels with different scales are convolved with the network data of different dimensions. Namely, a one-dimensional (1D) convolution kernel (n) is convolved with network data, and a two-dimensional (2D) convolution kernel $(n \times n)$ is convolved with a 2D matrix; then, the bias term b is added, and finally, the values are input to the activation function to obtain a series of feature maps. The mathematical expression of the convolution process is given by:

$$a_j^l = f\left(\sum\nolimits_{i \in m_j}\left(a_i^{l-1}\omega_{ij}^l\right) + b_j^l\right) \quad (1)$$

where a_j^l is the jth element of the lth layer; m_j is the jth convolution region of the feature map of the $(l-1)$th layer; a_i^{l-1} is the ith element in the $(l-1)$th layer, and ω_{ij}^l and b_j^l are the corresponding weight matrix and bias term, respectively; $f()$ is the activation function.

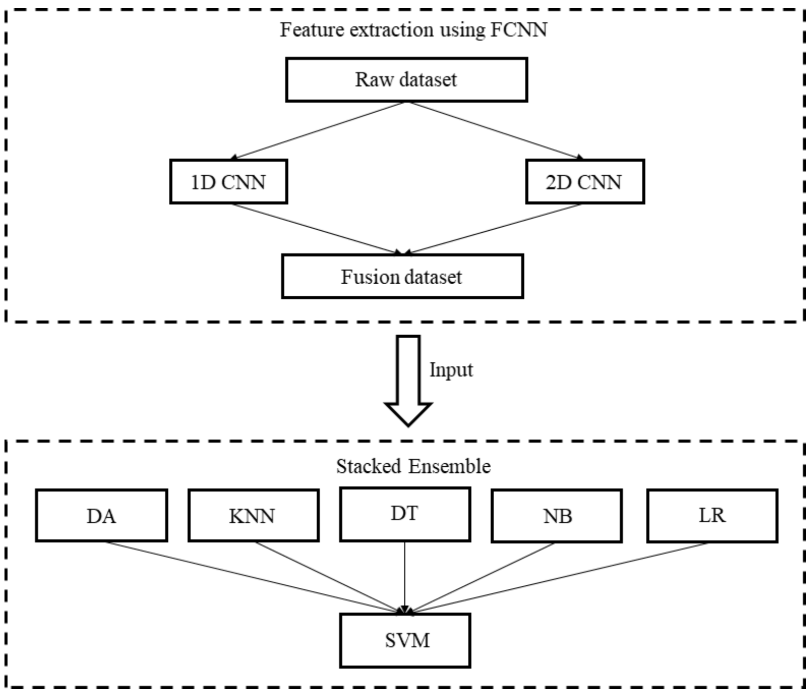


Figure 1. The FCNN-SE structure.

In this study, the ReLu activation function is used, and its mathematical expression is as follows [44]:

$$f(x) = \max[0, \lg(1 + e^x)] \tag{2}$$

To reduce the number of parameters required for CNN training and prevent overfitting and underfitting of a model and inadequate feature extraction, a pooling layer is added between successive convolutional layers. The mathematical expression of the pooling process is given by [44]:

$$a_j^l = f\left(\beta_j^l \text{down}\left(a_i^{l-1}\right) + b_j^l\right) \tag{3}$$

where down( ) is the downsampling function; β_j^l denotes the weight of the jth feature map in the lth layer; b_j^l denotes the bias term of the jth feature map in the lth layer.

The feature map after convolution pooling is fed to the fully connected layer for classification. The feature maps obtained by the convolution using different convolution kernels need to be flattened into one-dimensional vectors so that the local features can be integrated globally in higher dimensions.

In the backpropagation parameter update process of the FCNN, the error loss function is used to update network parameters, such as weights and biases of each layer, according to the error between the output value p_i and the expected value aiming to minimize the error between the output and expected values. In this study, the cross-entropy function is selected as an error cost function, and its mathematical expression is given by:

$$L = \frac{1}{N}\sum_i -[y_i \lg(p_i) + (1 - y_i)\lg(1 - p_i)] \tag{4}$$

where N denotes the number of samples, p_i is the training model's output value, and y_i is the expected value.

In the 1D and 2D convolution, the pooling layer is maximum pooling, and it is filled with SAME to ensure that the feature information will not be lost easily, and the Adam optimization algorithm is used to reduce the training time under the premise of achieving a more accurate data fitting. To obtain rich experience to extract features more effectively while avoiding overfitting caused by too many parameters, the size and number of convolutional kernels must be appropriate, so the same size convolutional kernels are selected in each dimension. By increasing the number of channels, the ability to extract feature information can be improved.

In this study, simulation experiments were conducted on the NSL-KDD dataset, which includes 41 features (An overview of all features is given in Appendix A). Since the data were 1D data, it could be directly input into the 1DCNN for processing. However, certain data processing was required before inputting the data in the 2DCNN. Eight more zeros were added to the data after 41 features to obtain 49 features, and then the 1D data were transformed into a 7 × 7 2D matrix, which was used as input data of the 2DCNN. The FCNN was used for feature extraction of the original dataset. Each of the two datasets obtained from the global maximum pooling layer of the 1DCNN and 2DCNN included 128 features, and these two datasets were combined, so a new dataset with 256 features was obtained. The specific parameters of the FCNN are shown in Table 1.

Table 1. The FCNN parameters.

No.	Layer	Size	Step	Number
1	Input layer Input layer	41 7 × 7	- -	- -
2	Convolution layer 1-1 Convolution layer 2-1	3 3 × 3	1 1 × 1	32 32
3	MaxPooling layer 1-1 MaxPooling layer 2-1	2 2 × 2	2 2 × 2	32 32
4	Convolution layer 1-2 Convolution layer 2-2	3 3 × 3	1 1 × 1	64 64
5	MaxPooling layer 1-2 MaxPooling layer 2-2	2 2 × 2	2 2 × 2	64 64
6	Convolution layer 1-3 Convolution layer 2-3	3 3 × 3	1 1 × 1	96 96
7	MaxPooling layer 1-3 MaxPooling layer 2-3	2 2 × 2	2 2 × 2	96 96
8	Convolution layer 1-4 Convolution layer 2-4	3 3 × 3	1 1 × 1	128 128
9	GlobalMaxPooling layer 1-1 GlobalMaxPooling layer 2-1	- -	- -	- -

3.2. SE Method

The SE method is shown in Figure 2. In the SR method, multiple base learners are trained using the original dataset. In the training process, the K-fold cross-validation is used to reduce the risk of model overfitting. The posterior probabilities of the base learners for each class of data are formed into a new dataset, which is used to retrain the meta-learner to obtain the final classification results.

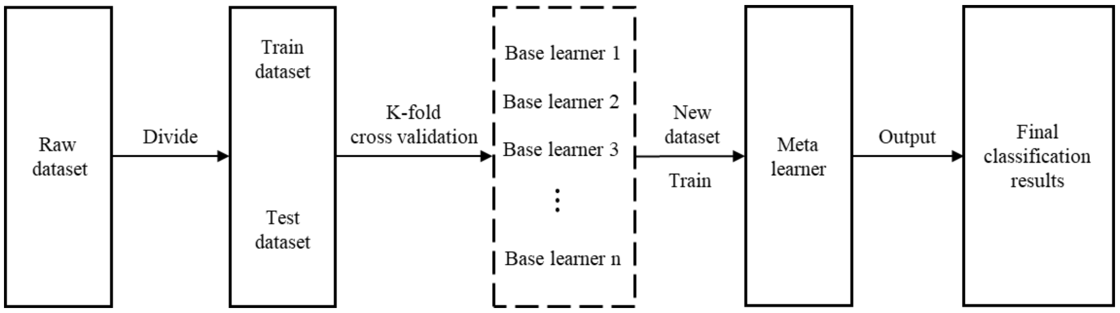


Figure 2. The SE method.

The specific steps of the SE algorithm are as follows:

1. Divide the original dataset into two parts: training set D, and test set T.
2. Perform the K-fold cross-validation of the base learners; randomly divide the original training set into K equal parts $(D_1, D_2, \cdots, D_k)$, where each base learner uses one of the parts as the K-fold test set and the remaining (K − 1) parts as the K-fold training set. Each base learner is trained using the K-fold training set, and the K-fold test set is used for classification. The posterior probabilities obtained by each base learner are combined and used as a training set $\widetilde{D}$ for the meta-learner.
3. Each base learner classifies the original test set T and uses the posterior probabilities as the test set $\widetilde{T}$ of the meta-learner.
4. The meta-learner uses the new dataset obtained from the base learners, incusing the training set $\widetilde{D}$ and test set $\widetilde{T}$, and performs learning and training, respectively, to output the final classification results.

The SE method uses the K-fold cross-validation to reduce the risk of model overfitting, while the posterior probabilities of multiple base learners are used for the second-stage training process. This method can overcome the limitations of a single learner; it combines the applicability and advantageous features of various learners and, thus, can improve the accuracy and generalization of classification results.

3.3. Radar Chart Method

The radar chart method, which is also known as integrated financial ratio analysis, allows simultaneous comparison of multiple factors, and it was originally developed for comprehensive analysis and evaluation of financial situations. Due to the remarkable characteristics of a simple and clear assessment process and intuitive results, in recent years, this method has been gradually applied to the evaluation and analysis of the power market and battery health [45,46]. In the radar chart method, first, the values of each basic indicator are standardized so that each indicator is transformed into a value with zero-mean and a variance of one to eliminate the quantitative differences between the indicators. In the quantitative analysis of a radar chart, the area and perimeter are usually selected as characteristic parameters. The larger the area of a radar chart is, the better the comprehensive performance of a model will be. When the radar chart area is certain, the smaller the perimeter is, the closer it will be to a circle, and the more the indicators will be likely to be equal, leading to the more balanced performance of the model from all aspects.

The area and perimeter of a radar chart are, respectively, defined by [47]:

$$\begin{cases} A_i = \sum\limits_{j=1}^{k} \frac{1}{2} n_{ij} n_{i(j+1)} \sin \alpha \\ L_i = \sum\limits_{j=1}^{k} \sqrt{n_{ij}^2 + n_{i(j+1)}^2 - 2 n_{ij} n_{j(j+1)} \cos \alpha} \end{cases} \tag{5}$$

where n_{ij} represents the jth evaluation indicator of the ith model, and $\alpha = \frac{2\pi}{k}$, where k is the number of indicators.

Construct the evaluation vector $v_i = [v_{i1}\ v_{i2}]$, where v_{i1} is the area evaluation value, and the larger this value is, the higher the comprehensive level of the evaluation object will be, and vice versa; v_{i2} is the perimeter evaluation value, which indicates the ratio to the area of a circle with the same perimeter; the larger the v_{i2} value is, the more balanced the evaluation object will be, and vice versa.

$$\begin{cases} v_{i1} = \frac{A_i}{\max A_i} \\ v_{i2} = \frac{A_i}{\left[\pi\left(\frac{L_i}{2\pi}\right)^2\right]} = \frac{4\pi A_i}{L_i^2} \end{cases} \tag{6}$$

In this study, the geometric mean method is used to construct the evaluation function F. The larger the value of this function is, the better the comprehensive performance of the model will be. The evaluation function is defined as follows:

$$F = \sqrt{v_{i1} v_{i2}} \tag{7}$$

Further, five basic evaluation indicators—accuracy, precision, recall, specificity, and F1 score—are selected as performance evaluation indexes. Since these five basic indicators are all dimensionless and are in a range of [0, 1], they do not need to be standardized and can be directly used in the evaluation process.

3.4. Basic Evaluation Indicators

The true-positive (*TP*), true-negative (*TN*), false-positive (*FP*), and false-negative (*FN*) values are calculated by comparing the true and predicted labels. The accuracy, precision, recall, specificity, and $F1$ score indicators are, respectively, defined by:

$$accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{8}$$

$$precision = \frac{TP}{TP + FP} \tag{9}$$

$$recall = \frac{TP}{TP + FN} \tag{10}$$

$$specificity = \frac{TN}{TN + FP} \tag{11}$$

$$F1\ score = \frac{2 \times precision \times recall}{precision + recall} \tag{12}$$

4. Results and Discussion

The operating system used in the experiment was Windows 10, and the simulation software MatlabR2021b was used. The hardware included 16 GB memory. The experiments were conducted on the NSL-KDD dataset, whose data were divided into two parts, normal and abnormal traffic data. The abnormal traffic data were further divided into four types: Probe, DoS, U2R, and R2L. The normal traffic and four types of abnormal traffic were used to construct five classes: Normal, Probe, DoS, U2R, and R2L. For recording purposes, the Normal, Probe, DoS, U2R, and R2L classes were labeled by 1, 2, 3, 4, and 5, respectively. Because the NSL-KDD dataset was an imbalanced dataset, the smote method was used to increase the proportion of minority class samples in the dataset.

4.1. Spearman Correlation Analysis

Five base learners, the DA, KNN, DT, NB, and LR, were used for intrusion detection classification, and the Spearman correlation coefficients between the base learners were calculated based on their classification results and presented in the form of a heat map, as shown in Figure 3. It was considered that if the Spearman correlation coefficient between two base learners was higher than 0.5, there was a high correlation between them; otherwise, there was a low correlation between base learners. As shown in Figure 3, the Spearman correlation coefficients of all base learners were all less than 0.5, indicating that the five base learners had a low correlation with each other on the NSL-KDD dataset. This was because the base learners adopted machine learning-based models with different mathematical construction principles and classification judgment rules, making it easy to show high variability. In addition, the intrusion detection dataset was different from the datasets used in other fields; the magnitudes and ranges between different features were different, and discrete and continuous data co-existed, making it easy for the models to show high variability.

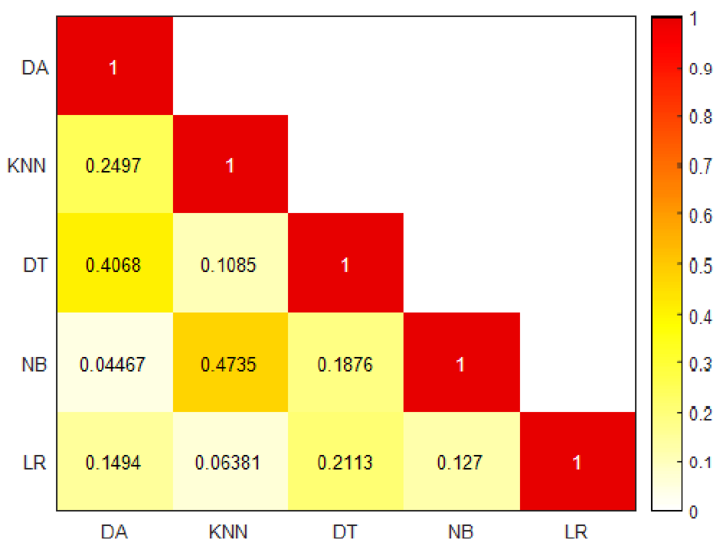


Figure 3. The Spearman correlation coefficient results.

4.2. Comprehensive Performance Evaluation based on Radar Chart Method

The proposed model FCNN-SE was compared with the DA, KNN, DT, NB, and LR base learners. The experimental results were expressed using comprehensive performance evaluation values based on the radar chart, as shown in Figure 4. See Table A2 for the index values. As shown in Figure 4 and Table A2, the FCNN-SE achieved the best results for all indicators only for class 4. On data of the other four classes, the FCNN-SE performance was not the best for all indicators. For instance, for class 1, the FCNN-SE had the highest accuracy and F1 score, the DT had the highest precision and specificity, and the LR had the highest recall. There was no single model that performed best on all indicators for classes 2, 3, and 5. These results show that the traditional evaluation method cannot compare the performance effectively. However, the comprehensive performance evaluation method based on the radar chart method can combine different indicators to provide a comprehensive comparison. The evaluation results obtained by the radar chart method showed that the comprehensive performance of the FCNN-SE was the best among all models for the five classes of data, indicating that the comprehensive performance of the FCNN-SE was better than those of the other comparison models.

4.3. McNemar Hypothesis Test Results

To verify the rationality of the introduced comprehensive performance evaluation method based on the radar chart method, the McNemar hypothesis test was performed using the FCNN-SE as a benchmark model. The McNemar hypothesis test confirmed the statistical significance of differences between the two methods. The McNemar hypothesis test is a nonparametric test on a 2×2 confusion matrix, and it is based on the standardized normal test statistic, which can be expressed as follows:

$$z = \frac{f_{12} - f_{21}}{\sqrt{f_{12} + f_{21}}} \tag{13}$$

where f_{ij} denotes the number of occurrences of elements (i, j) in the confusion matrix; the square of z obeys the chi-square distribution of one degree of freedom.

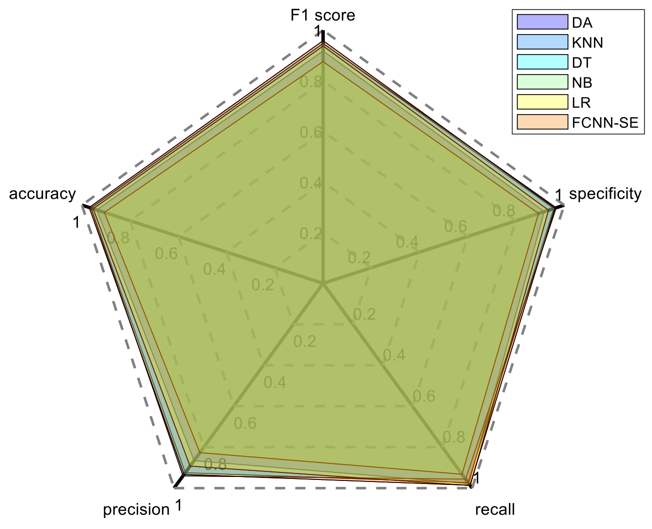
The test equation can be expressed by:

$$\chi^2 = \frac{(f_{12} - f_{21})^2}{f_{12} + f_{21}} \tag{14}$$

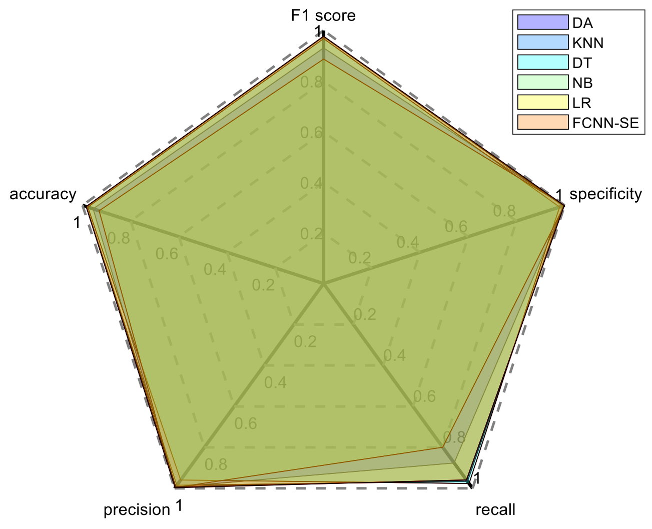
In the test, a $p < 0.05$ was considered to indicate statistical significance. The McNemar hypothesis test results of the comparison model and the benchmark model FCNN-SE are shown in Table 2. As shown in Table 2, the p values between the FCNN-SE and the other models were less than 0.05, indicating that the McNemar hypothesis was true. Thus, the detection capability of the FCNN-SE was statistically superior to those of the other base learners. This was consistent with the results obtained by the comprehensive performance evaluation method based on the radar chart method, which proves the correctness of the introduced comprehensive performance evaluation method based on the radar chart method.

Table 2. The McNemar hypothesis test results.

Model	**FCNN-SE** *p*-**Value**
DA	3.85×10^{-8}
KNN	3.16×10^{-3}
DT	1.70×10^{-3}
NB	4.58×10^{-9}
LR	5.88×10^{-6}

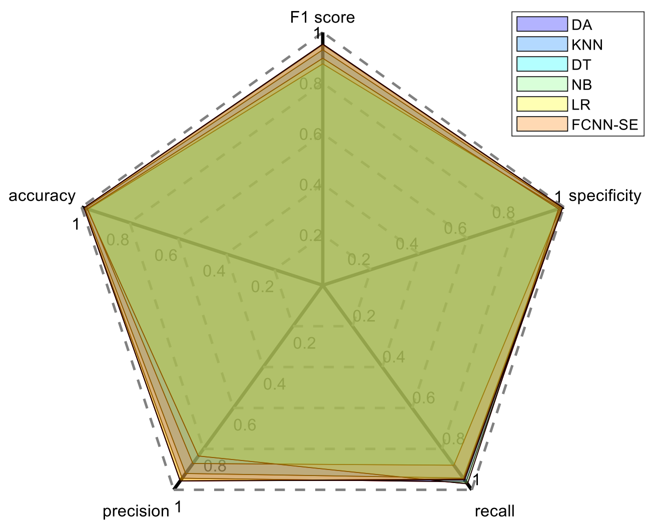


(**a**)

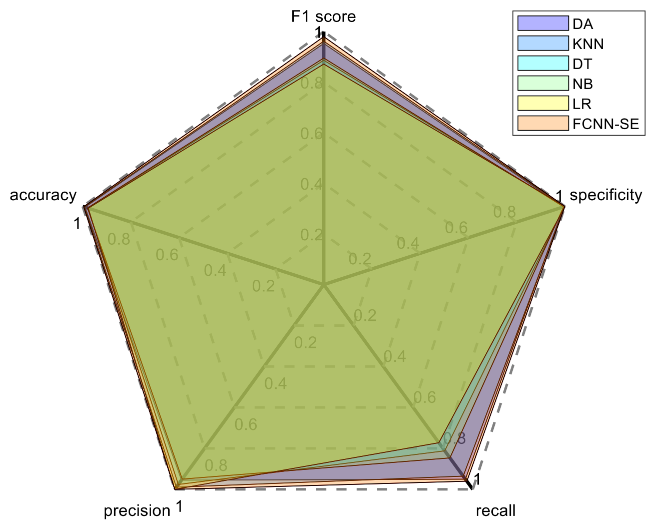


(**b**)

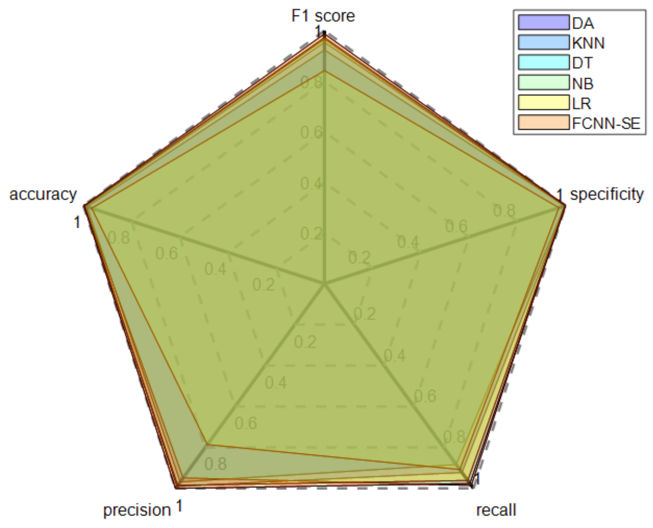
Figure 4. *Cont.*



(**c**)



(**d**)



(**e**)

Figure 4. The radar chart used for comprehensive model performance evaluation: (**a**) class 1 data; (**b**) class 2 data; (**c**) class 3 data; (**d**) class 4 data; (**e**) class 5 data.

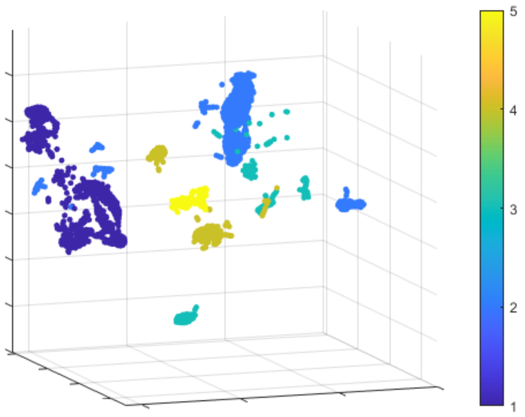
4.4. Ablation Experiment

To verify the superior performance of the FCNN-based feature extraction method, an ablation experiment was conducted on four datasets—the original data, the 1DCNN-processed data, the 2DCNN-processed data, and the FCNN-processed data. These data were used as input data of the SE model for classification detection. The comprehensive performance evaluation results are shown in Table 3, where it can be seen that the comprehensive performance evaluation result of the FCNN was the best among all models on all data except for the original data. Namely, the comprehensive performance evaluation result of the FCNN on the raw data was the lowest among all models, indicating that the dataset processed by the FCNN could better distinguish the differences between classes and improve model performance. Compared with the raw data, the comprehensive performance evaluation values of the 1DCNN and 2DCNN data were improved, but they were both lower than that of the FCNN. This was because the features extracted from a single dimension cannot fully express data information, which affects the classification performance of a model.

Table 3. Comprehensive performance evaluation results.

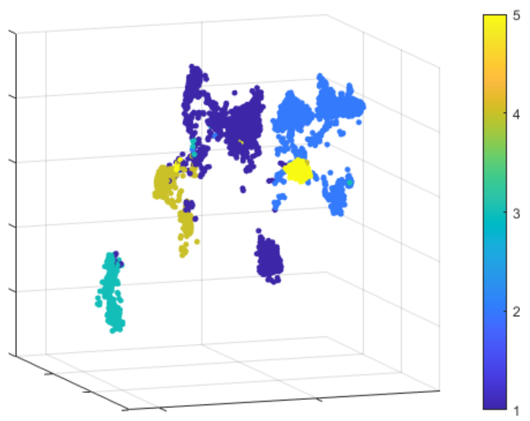
Model	Class 1	Class 2	Class 3	Class 4	Class 5
Raw Dataset	0.7957	0.7490	0.6262	0.7593	0.2679
1DCNN	0.9228	0.9138	0.9111	0.9172	0.8474
2DCNN	0.9264	0.9210	0.9277	0.9272	0.8206
FCNN	0.9296	0.9297	0.9296	0.9297	0.9299

To visualize the feature set and representation capability of features extracted by the FCNN, the visualization algorithm Barnes–Hut variation of t-SNE, which represents multi-dimensional features, was used to illustrate the distribution of samples in three dimensions; the results are shown in Figure 5. As shown in Figure 5a, there were obvious overlaps between the samples of different data types, and these overlaps occurred not only near the boundaries of the clusters but also within the clusters. Data of classes 2, 3, and 4 were divided into multiple regions. As shown in Figure 5b,c, compared with raw data, data samples processed by the 1DCNN and 2DCNN had a more concentrated distribution for the same type of data, outliers mostly disappeared, the class spacing was larger, and the overlap occurred only near the boundary between clusters. Further, as shown in Figure 5d, using the FCNN, the five types of data were completely separated, and there were only a few overlaps between classes 3 and 2, but their number was very small and thus could not affect the classification decision of the model. The results demonstrate that the FCNN-based feature extraction could effectively distinguish different types of data, which can provide a good basis for the performance improvement of the SE model.

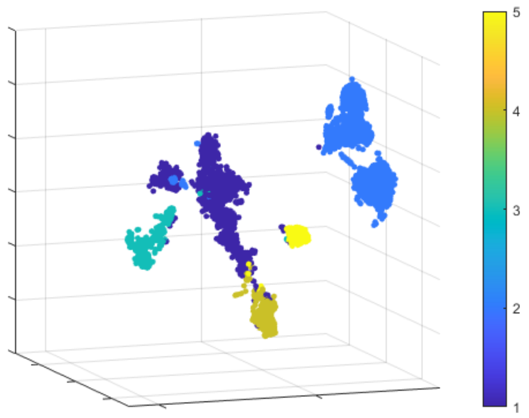


(**a**)

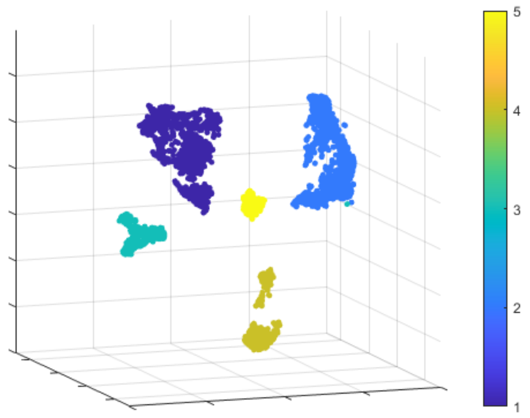
Figure 5. *Cont.*



(**b**)



(**c**)



(**d**)

Figure 5. The representation map of the Barnes–Hut variation of t-SNE: (**a**) raw data; (**b**) 1DCNN-processed data; (**c**) 2DCNN-processed data; (**d**) FCNN-processed data.

5. Conclusions

An intrusion detection model identifies malicious attacks by analyzing the characteristics of key nodes for network traffic and is an important part of the network security protection architecture. In this paper, an intrusion detection model based on the FCNN-SE is proposed. The proposed model can improve intrusion detection performance by combining heterogeneous base learners into classifiers and using the FCNN to extract feature information from traffic data. To measure the comprehensive performance of the proposed model, a comprehensive performance evaluation method based on the radar chart method is introduced to evaluate the experimental results. The in-depth experiments are conducted on the NSL-KDD dataset, and the experimental results demonstrated the FCNN-SE model is effective and can outperform the comparison models.

The limitations of this paper are as follows:

(1) Only used machine learning-based methods are used as base learners, while the neural network-based methods are ignored. Because the structure of the neural network itself is very complex, and multiple complex neural networks are integrated to work together at the same time, the structure of the model will be too large, which will greatly increase the training time of the model.

(2) The CNN is adopted as a base extractor without considering other feature extraction techniques. This paper has proved that CNN has good performance as a feature extractor, but the CNN designed in this paper is still very simple compared with mature neural networks such as GoogLeNet and ResNet, and cannot give full play to the powerful feature extraction ability of CNN.

In future work, we could try to lightweight the neural network design, and simplify the neural network structure on the basis of maintaining the excellent performance of the neural network. After joining the ensemble model, it will not make the ensemble model too large. At the same time, using the currently popular neural networks such as GoogLeNet and ResNet, and using transfer learning technology to extract higher-quality data traffic features.

Author Contributions: Conceptualization, C.C. and Y.S.; methodology, C.C.; software, S.Y.; validation, Y.S.; formal analysis, C.C.; resources, X.X.; writing—original draft preparation, L.Z., Q.L. and L.Y.; writing—review and editing, C.C.; supervision, C.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by National Natural Science Foundation of China (grant number: No.61703426, No.61806219, No.61876189), Youth talent promotion plan of Shaanxi University Association for science and technology (grant number: No. 20190108) and Shaanxi innovation capability support plan (grant number: No.2020KJXX-065).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All data used in this paper can be obtained by contacting the authors of this study.

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

Nomenclature

CNN	convolutional neural network
SE	stacked ensemble
FCNN	fusion CNN
1DCNN	one-dimensional CNN
2DCNN	two-dimensional CNN
NSL-KDD	a revised version of the KDD'99 dataset
DA	discriminant analysis
KNN	*K*-nearest neighbor
DT	decision tree
NB	naive Bayes
LR	logistic regression
SVM	support vector machine
TP	true positive
TN	true negative
FP	false positive
FN	false negative

Appendix A. An Overview of the Features in the NSL-KDD Dataset

The NSL-KDD dataset has 41 features, and their specific descriptions are shown in Table A1.

Table A1. Overview of the 41 features.

No.	Feature Name	Description	Type	Value Type	Ranges
1	Duration	Length of time duration of the connection	Continuous	Integers	0–54, 451
2	Protocol Type	Protocol used in the connection	Categorical	Strings	
3	Service	Destination network service used	Categorical	Strings	
4	Flag	Status of the connection—Normal or Error	Categorical	Strings	
5	Src Bytes	Number of data bytes transferred from source to destination in single connection	Continuous	Integers	0–1, 379, 963, 888
6	Dst Bytes	Number of data bytes transferred from destination to source in single connection	Continuous	Integers	0–30, 993, 7401
7	Land	If source and destination IP addresses and port numbers are equal then, this variable takes value 1 else 0	Binary	Integers	{0, 1}
8	Wrong Fragment	Total number of wrong fragments in this connection	Discrete	Integers	{0, 1, 3}
9	Urgent	Number of urgent packets in this connection. Urgent packets are packets with the urgent bit activated	Discrete	Integers	0–3
10	Hot	Number of "hot" indicators in the content such as: entering a system directory, creating programs and executing programs	Continuous	Integers	0–101
11	Num Failed Logins	Count of failed login attempts	Continuous	Integers	0–4
12	Logged In	Login Status: 1 if successfully logged in; 0 otherwise	Binary	Integers	{0, 1}
13	Num Compromised	Number of "compromised" conditions	Continuous	Integers	0–7479
14	Root Shell	1 if root shell is obtained; 0 otherwise	Binary	Integers	{0, 1}
15	Su Attempted	1 if "su root" command attempted or used; 0 otherwise	Discrete (Dataset contains '2' value)	Integers	0–2
16	Num Root	Number of "root" accesses or number of operations performed as a root in the connection	Continuous	Integers	0–7468
17	Num File Creations	Number of file creation operations in the connection	Continuous	Integers	0–100
18	Num Shells	Number of shell prompts	Continuous	Integers	0–2
19	Num Access Files	Number of operations on access control files	Continuous	Integers	0–9
20	Num Outbound Cmds	Number of outbound commands in an ftp session	Continuous	Integers	{0}
21	Is Hot Logins	1 if the login belongs to the "hot" lis, i.e., root or admin; else 0	Binary	Integers	{0, 1}
22	Is Guest Login	1 if the login is a "guest" login; 0 otherwise	Binary	Integers	{0, 1}
23	Count	Number of connections to the same destination host as the current connection in the past two seconds	Discrete	Integers	0–511
24	Srv Count	Number of connections to the same service (port number) as the current connection in the past two seconds	Discrete	Integers	0–511

Table A1. *Cont.*

No.	Feature Name	Description	Type	Value Type	Ranges
25	Serror Rate	The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in count (23)	Discrete	Floats (hundredths of a decimal)	0–1
26	Srv Serror Rate	The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in srv_count (24)	Discrete	Floats (hundredths of a decimal)	0–1
27	Rerror Rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in count (23)	Discrete	Floats (hundredths of a decimal)	0–1
28	Srv Rerror Rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in srv_count (24)	Discrete	Floats (hundredths of a decimal)	0–1
29	Same Srv Rate	The percentage of connections that were to the same service, among the connections aggregated in count (23)	Discrete	Floats (hundredths of a decimal)	0–1
30	Diff Srv Rate	The percentage of connections that were to different services, among the connections aggregated in count (23)	Discrete	Floats (hundredths of a decimal)	0–1
31	Srv Diff Host Rate	The percentage of connections that were to different destination machines among the connections aggregated in srv_count (24)	Discrete	Floats (hundredths of a decimal)	0–1
32	Dst Host Count	Number of connections having the same destination host IP address	Discrete	Integers	0–255
33	Dst Host Srv Count	Number of connections having the same port number	Discrete	Integers	0–255
34	Dst Host Same Srv Rate	The percentage of connections that were to different services, among the connections aggregated in dst_host_count (32)	Discrete	Floats (hundredths of a decimal)	0–1
35	Dst Host Diff Srv Rate	The percentage of connections that were to different services, among the connections aggregated in dst_host_count (32)	Discrete	Floats (hundredths of a decimal)	0–1
36	Dst Host Same Src Port Rate	The percentage of connections that were to the same source port, among the connections aggregated in dst_host_srv_count (33)	Discrete	Floats (hundredths of a decimal)	0–1
37	Dst Host Srv Diff Host Rate	The percentage of connections that were to different destination machines, among the connections aggregated in dst_host_srv_count (33)	Discrete	Floats (hundredths of a decimal)	0–1
38	Dst Host Serror Rate	The percentage of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in dst_host_count (32)	Discrete	Floats (hundredths of a decimal)	0–1
39	Dst Host Srv Serror Rate	The percent of connections that have activated the flag (4) s0, s1, s2 or s3, among the connections aggregated in dst_host_srv_count (33)	Discrete	Floats (hundredths of a decimal)	0–1
40	Dst Host Rerror Rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in dst_host_count (32)	Discrete	Floats (hundredths of a decimal)	0–1

Table A1. *Cont.*

No.	Feature Name	Description	Type	Value Type	Ranges
41	Dst Host Srv Rerror Rate	The percentage of connections that have activated the flag (4) REJ, among the connections aggregated in dst_host_srv_count (33)	Discrete	Floats (hundredths of a decimal)	0–1

Appendix B. Performance Comparison of Different Models

The proposed model FCNN-SE was compared with the DA, KNN, DT, NB, and LR base learners, and their performance comparison are shown in Table A2.

Table A2. Performance comparison between models.

Indicator	Model	Class 1	Class 2	Class 3	Class 4	Class 5
Accuracy	DA	0.9385	0.9560	0.9900	0.9905	0.9910
	KNN	0.9625	0.9835	0.9855	0.9925	0.9850
	DT	0.9630	0.9780	0.9740	0.9790	0.9930
	NB	0.9070	0.9310	0.9770	0.9800	0.9650
	LR	0.9535	0.9840	0.9900	0.9770	0.9945
	FCNN-SE	**0.9700**	**0.9850**	**0.9905**	**0.9960**	**0.9975**
Precision	DA	0.8657	0.9933	0.9434	0.9552	0.9896
	KNN	0.9246	0.9850	0.9202	0.9895	0.9681
	DT	**0.9401**	0.9594	0.8714	0.9762	0.9486
	NB	0.8266	0.9963	0.8347	0.9500	0.7857
	LR	0.8924	0.9924	0.9563	**1.0000**	0.9851
	FCNN-SE	0.9330	**0.9969**	**0.9565**	**1.0000**	**1.0000**
Recall	DA	0.9773	0.8761	0.9615	0.9505	0.9223
	KNN	0.9731	0.9661	0.9423	0.9356	0.8835
	DT	0.9561	**0.9764**	0.8798	0.8119	**0.9854**
	NB	0.9320	0.7994	**0.9712**	0.8465	0.9078
	LR	**0.9873**	0.9602	0.9471	0.7723	0.9612
	FCNN-SE	0.9858	0.9587	0.9519	**0.9604**	0.9757
Specificity	DA	0.9173	0.9970	0.9933	0.9950	0.9989
	KNN	0.9567	0.9924	0.9905	0.9989	0.9967
	DT	**0.9668**	0.9788	0.9849	0.9978	0.9939
	NB	0.8934	**0.9985**	0.9777	0.9950	0.9716
	LR	0.9351	0.9962	**0.9950**	**1.0000**	0.9983
	FCNN-SE	0.9614	**0.9985**	**0.9950**	**1.0000**	**1.0000**
F1 score	DA	0.9182	0.9310	0.9524	0.9529	0.9548
	KNN	0.9482	0.9754	0.9311	0.9618	0.9239
	DT	0.9480	0.9678	0.8756	0.8865	0.9667
	NB	0.8762	0.8871	0.8978	0.8953	0.8423
	LR	0.9375	0.9760	0.9517	0.8715	0.9730
	FCNN-SE	**0.9587**	**0.9774**	**0.9542**	**0.9798**	**0.9877**
Comprehensive performance evaluation value	DA	0.8910	0.8960	0.9281	0.9121	0.9093
	KNN	0.9212	0.9272	0.9138	0.9184	0.8892
	DT	0.9232	0.9193	0.8758	0.8691	0.9158
	NB	0.8559	0.8635	0.8891	0.8748	0.8310
	LR	0.9087	0.9283	0.9279	0.8589	0.9205
	FCNN-SE	**0.9296**	**0.9297**	**0.9296**	**0.9297**	**0.9299**

References

1. Samriya, J.K.; Tiwari, R.; Cheng, X.; Singh, R.K.; Shankar, A.; Kumar, M. Network intrusion detection using ACO-DNN model with DVFS based energy optimization in cloud framework. *Sustain. Comput. Inform. Syst.* **2022**, *35*, 100746. [CrossRef]
2. Imran, M.; Khan, S.; Hlavacs, H.; Alam Khan, F.; Anwar, S. Intrusion detection in networks using cuckoo search optimization. *Soft Comput.* **2022**. [CrossRef]

3. You, L.; Wang, Z. A Cloud Based Network Intrusion Detection System. *Teh. Vjesn.* **2022**, *29*, 987–992.
4. Tsimenidis, S.; Lagkas, T.; Rantos, K. Deep learning in iot intrusion detection. *J. Netw. Syst. Manag.* **2022**, *30*, 8. [CrossRef]
5. Mills, R.; Marnerides, A.K.; Broadbent, M.; Race, N. Practical Intrusion Detection of Emerging Threats. *IEEE Trans. Netw. Serv. Manag.* **2021**, *19*, 582–600. [CrossRef]
6. Wang, Z.; Shao, L.; Cheng, K.; Liu, Y.; Jiang, J.; Nie, Y.; Li, X.; Kuang, X. ICDF: Intrusion collaborative detection framework based on confidence. *Int. J. Intell. Syst.* **2022**. [CrossRef]
7. Deolindo, V.M.; Dalmazo, B.L.; da Silva, M.V.; de Oliveira, L.R.; Silva, A.D.B.; Granville, L.Z.; Gaspary, L.P.; Nobre, J.C. Using Quadratic Discriminant Analysis by Intrusion Detection Systems for Port Scan and Slowloris Attack Classification. In Proceedings of the International Conference on Computational Science and Its Applications, Cagliari, Italy, 13–16 September 2021; Springer: Cham, Switzerland, 2021.
8. Liu, G.; Zhao, H.; Fan, F.; Liu, G.; Xu, Q.; Nazir, S. An Enhanced Intrusion Detection Model Based on Improved kNN in WSNs. *Sensors* **2022**, *22*, 1407. [CrossRef]
9. Das, A.; Sunitha, B.S. An Efficient Feature Selection Approach for Intrusion Detection System using Decision Tree. *Int. J. Adv. Comput. Sci. Appl.* **2022**, *13*. [CrossRef]
10. Singh, S. Poly Logarithmic Naive Bayes Intrusion Detection System Using Linear Stable PCA Feature Extraction. *Wirel. Pers. Commun.* **2022**, *12*, 3117–3132. [CrossRef]
11. Kanimozhi, P.; Victoire, T.A.A. Oppositional tunicate fuzzy C-means algorithm and logistic regression for intrusion detection on cloud. *Concurr. Comput. Pract. Exp.* **2021**, *34*, e6624. [CrossRef]
12. Chen, C.; Liu, S.; Wang, Y.; Zhu, Y. A Network Intrusion Detection Method Based on PSOGWO-SVM. *J. Air Force Eng. Univ. Nat. Sci. Ed.* **2022**, *23*, 97–105.
13. Li, Y.; Xu, W.; Li, W.; Li, A.; Liu, Z. Research on hybrid intrusion detection method based on the ADASYN and ID3 algorithms. *Math. Biosci. Eng.* **2021**, *19*, 2030–2042. [CrossRef] [PubMed]
14. Hassan, M.; Haque, E.; Tozal, M.E.; Raghavan, V.; Agrawal, R. Intrusion Detection Using Payload Embeddings. *IEEE Access* **2021**, *10*, 4015–4030. [CrossRef]
15. Wang, Z.; Schapire, R.E.; Verma, N. Error adaptive classifier boosting (EACB): Leveraging data-driven training towards hardware resilience for signal inference. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2015**, *62*, 1136–1145. [CrossRef]
16. Creamer, G.; Freund, Y. Using boosting for financial analysis and performance prediction: Application to S&P 500 companies, Latin American ADRs and banks. *Comput. Econ.* **2010**, *36*, 133–151.
17. Breiman, L. Using iterated bagging to debias regressions. *Machine Learning Mach. Learn.* **2001**, *45*, 261–277. [CrossRef]
18. Tang, Y.; Gu, L.; Wang, L. Deep Stacking Network for Intrusion Detection. *Sensors* **2021**, *22*, 25. [CrossRef]
19. Basati, A.; Faghih, M.M. DFE: Efficient IoT network intrusion detection using deep feature extraction. *Neural Comput. Appl.* **2022**, *34*, 15175–15195. [CrossRef]
20. Fatani, A.; Dahou, A.; Al-Qaness, M.A.A.; Lu, S.; Elaziz, M.A. Advanced feature extraction and selection approach using deep learning and Aquila optimizer for IoT intrusion detection system. *Sensors* **2021**, *22*, 140. [CrossRef]
21. Cui, J.; Zong, L.; Xie, J.; Tang, M. A novel multi-module integrated intrusion detection system for high-dimensional imbalanced data. *Appl. Intell.* **2022**, 1–17. [CrossRef]
22. Cao, B.; Li, C.; Song, Y.; Qin, Y.; Chen, C. Network Intrusion Detection Model Based on CNN and GRU. *Appl. Sci.* **2022**, *12*, 4184. [CrossRef]
23. Bhuvansehwari, K.S. Improved Dragonfly Optimizer for Intrusion Detection Using Deep Clustering CNN-PSO Classifier. *CMC-Comput. Mater. Contin.* **2022**, *70*, 5949–5965. [CrossRef]
24. Zhang, H.; Li, J.-L.; Liu, X.-M.; Dong, C. Multi-dimensional feature fusion and stacking ensemble mechanism for network intrusion detection. *Futur. Gener. Comput. Syst.* **2021**, *122*, 130–143. [CrossRef]
25. Qiu, W.; Ma, Y.; Chen, X.; Yu, H.; Chen, L. Hybrid intrusion detection system based on Dempster-Shafer evidence theory. *Comput. Secur.* **2022**, *117*, 102709. [CrossRef]
26. Mehmood, M.; Javed, T.; Nebhen, J.; Abbas, S.; Abid, R.; Bojja, G.R.; Rizwan, M. A Hybrid approach for network intrusion detection. *CMC-Comput. Mater. Contin.* **2022**, *70*, 91–107z. [CrossRef]
27. Kim, T.; Pak, W. Real-time network intrusion detection using deferred decision and hybrid classifier. *Futur. Gener. Comput. Syst.* **2022**, *132*, 51–66. [CrossRef]
28. Qiang, D.; Zhang, L.; Huang, X. Quantitative evaluation of TOD performance based on multi-source data: A case study of Shanghai. *Front. Public Health* **2022**, *10*, 820694. [CrossRef]
29. Yang, M.; Ji, Z.; Zhang, L.; Zhang, A.; Xia, Y. A hybrid comprehensive performance evaluation approach of cutter holder for tunnel boring machine. *Adv. Eng. Inform.* **2022**, *52*, 101546. [CrossRef]
30. Abushark, Y.B.; Khan, A.I.; Alsolami, F.; Almalawi, A.; Alam, M.; Agrawal, A.; Kumar, R.; Khan, R.A. Cyber Security Analysis and Evaluation for Intrusion Detection Systems. *CMC-Comput. Mater. Contin.* **2022**, *72*, 1765–1783. [CrossRef]
31. Zheng, D.; Hong, Z.; Wang, N.; Chen, P. An improved LDA-based ELM classification for intrusion detection algorithm in IoT application. *Sensors* **2020**, *20*, 1706. [CrossRef]
32. Labiod, Y.; Korba, A.A.; Ghoualmi, N. Fog Computing-Based Intrusion Detection Architecture to Protect IoT Networks. *Wirel. Pers. Commun.* **2022**, *125*, 231–259. [CrossRef]

33. Saba, T.; Rehman, A.; Sadad, T.; Kolivand, H.; Bahaj, S.A. Anomaly-based intrusion detection system for IoT networks through deep learning model. *Comput. Electr. Eng.* **2022**, *99*, 107810. [CrossRef]
34. Yu, J.; Ye, X.; Li, H. A high precision intrusion detection system for network security communication based on multi-scale convolutional neural network. *Futur. Gener. Comput. Syst.* **2022**, *129*, 399–406. [CrossRef]
35. Wu, Z.; Gao, P.; Cui, L.; Chen, J. An incremental learning method based on dynamic ensemble RVM for intrusion detection. *IEEE Trans. Netw. Serv. Manag.* **2021**, *19*, 671–685. [CrossRef]
36. Mokbal, F.; Dan, W.; Osman, M.; Ping, Y.; Alsamhi, S. An efficient intrusion detection framework based on embedding feature selection and ensemble learning technique. *Int. Arab. J. Inf. Technol.* **2022**, *19*, 237–248. [CrossRef]
37. Alanazi, M.; Aljuhani, A. Anomaly detection for internet of things cyberattacks. *Comput. Mater. Contin.* **2022**, *72*, 261–279. [CrossRef]
38. Prasad, M.; Gupta, R.K.; Tripathi, S. A Multi-level Correlation-Based Feature Selection for Intrusion Detection. *Arab. J. Sci. Eng.* **2022**, *47*, 10719–10729. [CrossRef]
39. Patil, D.; Pattewar, T. Majority Voting and Feature Selection Based Network Intrusion Detection System. *ICST Trans. Scalable Inf. Syst.* **2018**, e48. [CrossRef]
40. Quincozes, S.E.; Passos, D.; Albuquerque, C.; Mossé, D.; Ochi, L.S. An extended assessment of metaheuristics-based feature selection for intrusion detection in CPS perception layer. *Ann. Telecommun.* **2022**, *77*, 457–471. [CrossRef]
41. Prakash, P.J.; Lalitha, B. Optimized Ensemble Classifier Based Network Intrusion Detection System for RPL Based Internet of Things. *Wirel. Pers. Commun.* **2022**, *125*, 3603–3626. [CrossRef]
42. Babu, R.A.; Kannan, S. Bat-Inspired Optimization for Intrusion Detection Using an Ensemble Forecasting Method. *Intell. Autom. Soft Comput.* **2022**, *34*, 307–323. [CrossRef]
43. Niu, Y.; Chen, C.; Zhang, X.; Zhou, X.; Liu, H. Application of a New Feature Generation Algorithm in Intrusion Detection System. *Wirel. Commun. Mob. Comput.* **2022**, *2022*, 3794579. [CrossRef]
44. Wang, Y.; Jin, Z. Rolling bearing fault diagnosis based on fusion CNN and PSO-SVM. *J. Mech. Strength* **2021**, *43*, 793–797.
45. Cui, J.; Li, Y.; Lin, Z.; He, C.; Wang, P.; Li, Y.; Liu, X.; Zhang, Z.; Qian, H.; Lin, Z.; et al. Multi-dimensional evaluation of power market based on multiple attribute decision making. *Energy Rep.* **2022**, *8*, 59–65. [CrossRef]
46. Ge, D.; Zhang, Z.; Kong, X.; Wan, Z. Extreme Learning Machine Using Bat Optimization Algorithm for Estimating State of Health of Lithium-Ion Batteries. *Appl. Sci.* **2022**, *12*, 1398. [CrossRef]
47. Du, X.; Teng, G.; Du, X.; Liu, M.L.; Wang, C.Y. Comprehensive evaluation of environmental comfort in layer poultry house using radar graph. *Trans. Chin. Soc. Agric. Eng.* **2020**, *36*, 202–209.