EAI Endorsed Transactions
on Scalable Information Systems

Research Article EAI.EU

NAS-FD: Neural Architecture Search-Based Fraud Detection for Power Audit Data

Yuanzhong Zuo [1], Jingyi Hu [2], Rao Kuang [2], Ping Chen [3,4*], Zhigao Zheng [5], Ze Zhou [6]

[1] State Grid Hubei Electric Power Co., Ltd., Wuhan 430048, China
[2] Audit and Supervision Department, State Grid Hubei Electric Power Co., Ltd., Wuhan 430048, China
[3] Technical Training Center, State Grid Hubei Electric Power Co., Ltd., Wuhan 430048, China
[4] Wuhan Electric Power Technical College, Wuhan 430074, China
[5] School of Computer Science, Wuhan University, Wuhan 430074, China
[6] School of Automation, Wuhan University of Technology, Wuhan 430070, China

Abstract

Power data auditing is the cornerstone of a reliable and efficient modern power system. Various deep learning models have been successfully applied to fraud detection in power audit data. However, most of these methods rely on manual trial-and-error and expert knowledge to design the neural architectures and hyper-parameters. To address this limitation, this paper proposes an innovative automated deep learning approach for fraud detection model design using genetic algorithm (GA)-based neural architecture search (NAS), termed NAS-FD. In NAS-FD, a convolutional neural network (CNN) is employed as the primary detection model, leveraging its strong data learning and feature extraction capabilities. First, an effective encoding scheme is developed to represent the neural architectures and hyper-parameters of CNN, as these parameters significantly influence the detection performance. Then, considering detection performance as the objective function, well-designed GA-based evolutionary operations are implemented to optimize the neural architectures and hyper-parameters of CNN, obtaining the optimized CNN. The detection performance of the proposed NAS-FD method is validated using an electricity theft dataset from the power auditing domain. Experimental results demonstrate that NAS-FD achieves superior detection performance compared with manually designed deep learning models in terms of four performance indices including accuracy, precision, recall, and F1-score.

Received on 10 Nov, 2025; accepted on 10 Jan, 2026; published on 14 January 2026

Keywords: Modern power systems, power audit data, fraud detection, convolutional neural network, neural architecture search

Copyright © 2026 Yuanzhong Zuo *et al.*, licensed to EAI. This is an open access article distributed under the terms of the Creative Commons Attribution license (http://creativecommons.org/licenses/by/4.0/), which permits unlimited use, distribution and reproduction in any medium so long as the original work is properly cited.

doi:10.4108/eetsis.10860

1. Introduction

Modern power systems serve as the cornerstone of national critical infrastructure, whose secure, stable, and economic operation is directly linked to national welfare and energy security [1]. Power auditing plays a vital role as both a supervisor and a diagnostician in this framework. It provides comprehensive insights and evaluations of system operational status, asset efficiency, power quality, and market transactions through the continuous collection and analysis of massive measurement data[2], [3]. Effective auditing is not only crucial for ensuring the economic operation of the grid, optimizing resource allocation, and improving operational efficiency, but also constitutes the first line of defense for timely fraud detection, potential risk identification, and precise decision-making support. Among the various dimensions of power auditing, electricity theft detection holds exceptional importance due to the significant direct economic losses and serious threats to grid security it presents. Electricity theft not only causes substantial annual financial losses for utility companies but can also lead to local line overloads, voltage instability, abnormal transformer

[*] Corresponding author. Email: chenp35@hb.sgcc.com.cn

aging, and even fires or safety incidents from malicious tampering, severely disrupting the normal operation of the distribution network [4], [5].

Traditional methods for fraud detection and electricity theft identification in power auditing predominantly rely on periodic on-site inspections, user complaints, or simple threshold-based statistical analysis. These approaches suffer from low efficiency, limited coverage, and high vulnerability to evasion. Consequently, the development of advanced detection algorithms capable of automatically, accurately, and promptly identifying concealed and evolving electricity theft patterns from massive, high-dimensional, and heterogeneous power data has become an urgent and highly valuable research topic in power system auditing[6], [7]. Artificial intelligence (AI), encompassing machine learning and deep learning, offers significant advantages in data mining and feature extraction. The detection performance for anomalous power audit data has been substantially enhanced through the application of various AI models, including artificial neural networks (ANNs) [8], clustering methods [5], convolutional neural networks (CNNs) [9], and long short-term memory (LSTM) model [10]. Given its capability to automatically learn and extract complex spatial features from raw network traffic data without relying on expert knowledge for manual feature engineering, CNN is adopted as the core detection model in this work. Furthermore, its strong ability to recognize local patterns enables effective detection of unknown and variant fraud in power audit data.

Although deep learning-based methods have been successfully applied in fraud detection for power auditing, their detection performance is highly dependent on the model neural architecture and hyper-parameters. Most current approaches rely on manual trial-and-error adjustment, which is time-consuming[9], [10]. To automate the tuning of neural architectures, neural architecture search (NAS) has gained significant popularity. NAS designs encoding methods, defines objective functions, and employs evolutionary search to ultimately obtain optimized models for different tasks [11], [12]. For example, Real *et al.* [13] employed simple evolutionary techniques with novel mutation operators to optimize both neural architectures and hyper-parameters of CNNs for image classification. Similarly, Sun *et al.* [14] utilized a genetic algorithm (GA)-based method to search the optimized neural architectures, initial weights, and activation functions in an unsupervised deep learning model, evaluated on MNIST and CIFAR-10. In a different application, Ho *et al.* [15] introduced a GA-NAS method with binary representation to optimize encoder-decoder architectures and meta-parameters for deep image prior. In [16], a novel NAS framework was presented, which optimizes the neural architectures and hyper-parameters of a variational autoencoder for unsupervised anomaly detection in the Internet-of-Things (IoT) security domain. A hybrid deep learning model, comprising a combination of CNN and recurrent neural networks, is employed to extract features from IoT datasets for accurate intrusion detection. The neural architectures and hyper-parameters of this hybrid model are optimized using particle swarm optimization [17]. Dong et al. [18], and Fei et al. [19] apply an Auto-Keras automatic detection framework, combining NAS and Bayesian optimization, to non-technical loss detection in distribution electricity networks. The framework automatically searches and integrates models on preprocessed electricity consumption data, achieving performance comparable to manually designed models without requiring professional expertise. While NAS requires computational time for the architecture exploration, it offers the advantage of automation over manual trial-and-error approaches, enabling systematic convergence toward an optimal architecture. Moreover, the process is conducted offline. Once the optimal architecture is identified, its deployment in online systems incurs no additional search overhead.

The aforementioned experimental results demonstrate that employing NAS technology yields superior performance compared to manual trial-and-error methods. However, most existing methods do not consider all the hyperparameters that can be optimized within the neural network, and the chosen evolutionary algorithms have not been thoroughly validated for their optimization capability. As a result, the performance of NAS is severely limited. Moreover, existing NAS-based deep learning models primarily focus on image classification tasks or cyber attack detection in internet domains[13–17]. Due to the distinct characteristics of power audit data, which differ significantly from these domains, directly applying these methods is challenging. Consequently, a redesigned NAS approach is required for addressing fraud detection in power audit data. The NSA design problem is inherently non-differentiable and non-convex. As a prominent evolutionary algorithm, GA is well-suited for such challenges as it imposes no specific constraints on the problem landscape, exhibits low sensitivity to initial solutions, and is effective for complex real-world optimization. Owing to these advantages, GA is a promising approach for automating CNN design [14],[15].

These observations strongly motivate an automated fraud detection framework for power audit data that combines the advantages of NAS with CNN-specific characteristics. Consequently, a novel fraud detection method termed NAS-FD is proposed, which integrates the strengths of GA-based NAS and CNNs. The main contributions of this paper are as follows:

(1) Most fraud detection models for power audit data rely on manually designed deep learning models through trial and error, which demands substantial

human and computational resources and heavily depends on expert knowledge. This paper develops a NAS-based fraud detection method using genetic algorithms, termed NAS-FD, which automates the design of CNN neural architectures and hyper-parameters.

(2) From an optimization perspective, the fraud detection model for power audit data is formulated as a single-objective optimization problem. A hybrid encoding scheme is employed to represent the neural architectures and hyper-parameters of CNN. Subsequently, well-designed GA-based evolutionary operations, including crossover operation and mutation operation, are applied to evolve these parameters to find the optimized CNN.

(3) To validate the effectiveness of the proposed NAS-FD method, an electricity theft dataset in the power audit domain, i.e., the theft detection dataset (TDD2022) [8] is utilized. Experimental results demonstrate that NAS-FD achieves superior detection performance according to four common performance indices including accuracy, precision, recall, and F1-score by comparing with manually designed CNN and LSTM.

The remainder of this paper is organized as follows. Section II provides the preliminaries including CNN and GA. Section III presents the proposed NAS-FD method. Experimental results and analysis are provided in Section IV. Finally, Section V concludes the paper.

2. Preliminaries

Before introducing the proposed NAS-FD method for power audit data, this section briefly presents the preliminaries on CNN and GA.

2.1. Convolutional neural networks

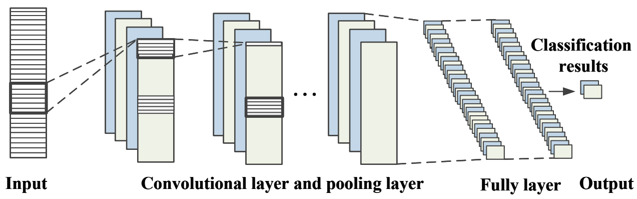


Figure 1. The structure of CNN for the classification task.

Fig. 1 gives the structure of CNN for the classification task. CNN is a specialized class of deep learning model designed to process sequential data with a topological structure, such as time-series signals, audio waveforms, and text sequences. Their architecture is particularly effective at extracting local, translation-invariant features through the core operations of convolution and pooling [20], [21].

The fundamental building block of a CNN is the convolutional layer. This layer applies a set of filters to the input sequence by sliding them across the temporal dimension. Each filter is responsible for detecting a specific local pattern or feature at different positions in the sequence. The convolution operation between a 1D input signal $\mathbf{x} \in \mathbb{R}^L$ of length L and a filter $\mathbf{w} \in \mathbb{R}^k$ of size k at position i of the output feature map is mathematically defined as:

$$(\mathbf{x} * \mathbf{w})[i] = \sum_{j=0}^{k-1} \mathbf{w}[j] \cdot \mathbf{x}[i+j] \tag{1}$$

where $*$ denotes the convolution operator. To control the size of the output feature map and to increase the receptive field, a stride s and zero-padding p are often introduced. The length of the output feature map L' can be calculated as:

$$L' = \left\lfloor \frac{L + 2p - k}{s} \right\rfloor + 1 \tag{2}$$

Subsequently, a non-linear activation function is applied element-wise to the convolution output to introduce non-linearity into the model, enabling it to learn complex patterns. The final output of a convolutional layer with F filters is a set of F feature maps, denoted as $\mathbf{h}$:

$$\mathbf{h} = \sigma(\mathbf{W} * \mathbf{x} + \mathbf{b}) \tag{3}$$

where $\mathbf{W}$ is the tensor of filters, $\mathbf{b}$ is the bias vector. σ is the activation function.

Following the convolutional layers, pooling layers are typically employed to reduce the dimensionality of the feature maps, thereby decreasing the computational cost and number of parameters, while also providing a form of translation invariance. The most common operation is max-pooling, which outputs the maximum value within a sliding window of size p_s as follows:

$$y_i = \max_{(i-1)p_s+1 \le t \le i \cdot p_s} h_t \tag{4}$$

After several stacks of convolution and pooling layers, the high-level features are flattened and passed to one or more fully-connected layers for the final classification task.

In summary, CNNs leverage local connectivity, weight sharing, and hierarchical feature learning to achieve state-of-the-art performance on a classification task. In fraud detection, the advantage of CNN lies in its ability to efficiently extract local features and discriminative patterns directly from raw one-dimensional sequential data without relying on complex preprocessing or manual feature engineering. The convolutional kernels sliding along the temporal dimension effectively capture local temporal dependencies and short-term abnormal morphologies. Moreover, its hierarchical structure enables automatic learning of multi-scale feature representations. Additionally, CNN offers greater advantages in computational efficiency and model complexity, making it particularly suitable for real-time or resource-constrained detection scenarios.

2.2. Genetic algorithm

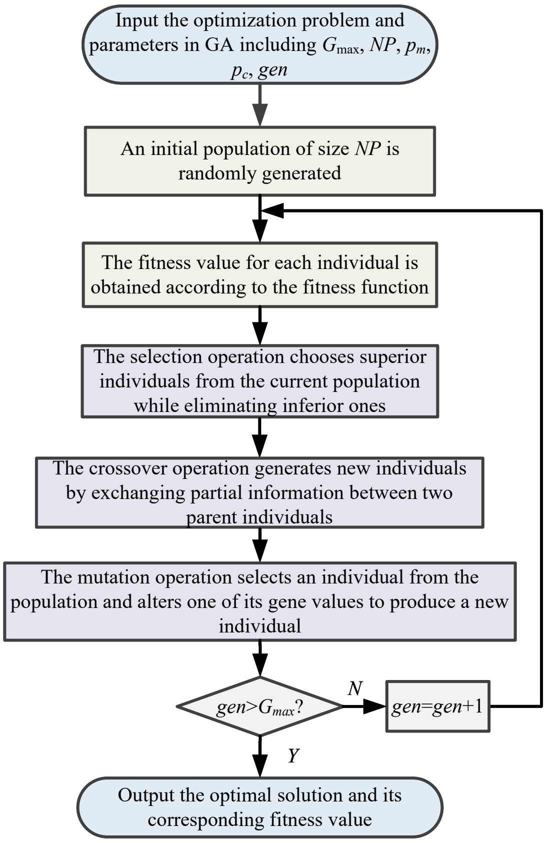

Input the optimization problem and parameters in GA including $G_{\max}$, NP, p_m, p_c, gen
An initial population of size NP is randomly generated
The fitness value for each individual is obtained according to the fitness function
The selection operation chooses superior individuals from the current population while eliminating inferior ones
The crossover operation generates new individuals by exchanging partial information between two parent individuals
The mutation operation selects an individual from the population and alters one of its gene values to produce a new individual
$gen>G_{max}$?
N
$gen=gen+1$
Y
Output the optimal solution and its corresponding fitness value

Figure 2. The flowchart of the classical GA.

The GA is an evolutionary algorithm simulating natural evolution processes, developed as a search algorithm for optimized solutions inspired by natural selection, crossover, and mutation phenomena [22]. In NAS, GA is valued for its suitability in parallel global optimization capability, and flexibility in handling complex optimization problems. GA-based evolutionary mechanism balances exploration and exploitation, circumventing the reliance on continuous differentiable spaces required by gradient-based methods, making it particularly effective for high-dimensional optimization of NAS. GA contains some advantages such as simple principles, strong generality, and high robustness. Fig. 2 shows the flowchart of the classical GA, with the corresponding basic steps provided as follows.

Input: Complex optimization problem, i.e., the fitness function, and GA parameters including maximum iterations $G_{\max}$, population size NP, crossover probability p_c, mutation probability p_m, and iteration counter gen.

Output: The best individual and its corresponding fitness value.

Step 1: Perform population initialization. Randomly generate an initial population of size NP, where each individual represents a feasible solution in the search space. Set $gen = 1$.

Step 2: Perform fitness evaluation. Compute the fitness value for each individual according to the fitness function.

Step 3: Perform selection operation. Select superior individuals from the current population based on fitness values, eliminating inferior ones. For minimization problems, individuals with smaller fitness values are preferred. The opposite holds for maximization problems.

Step 4: Perform crossover operation. Generate new individuals by exchanging partial information between two parent individuals via crossover, enabling gene information exchange.

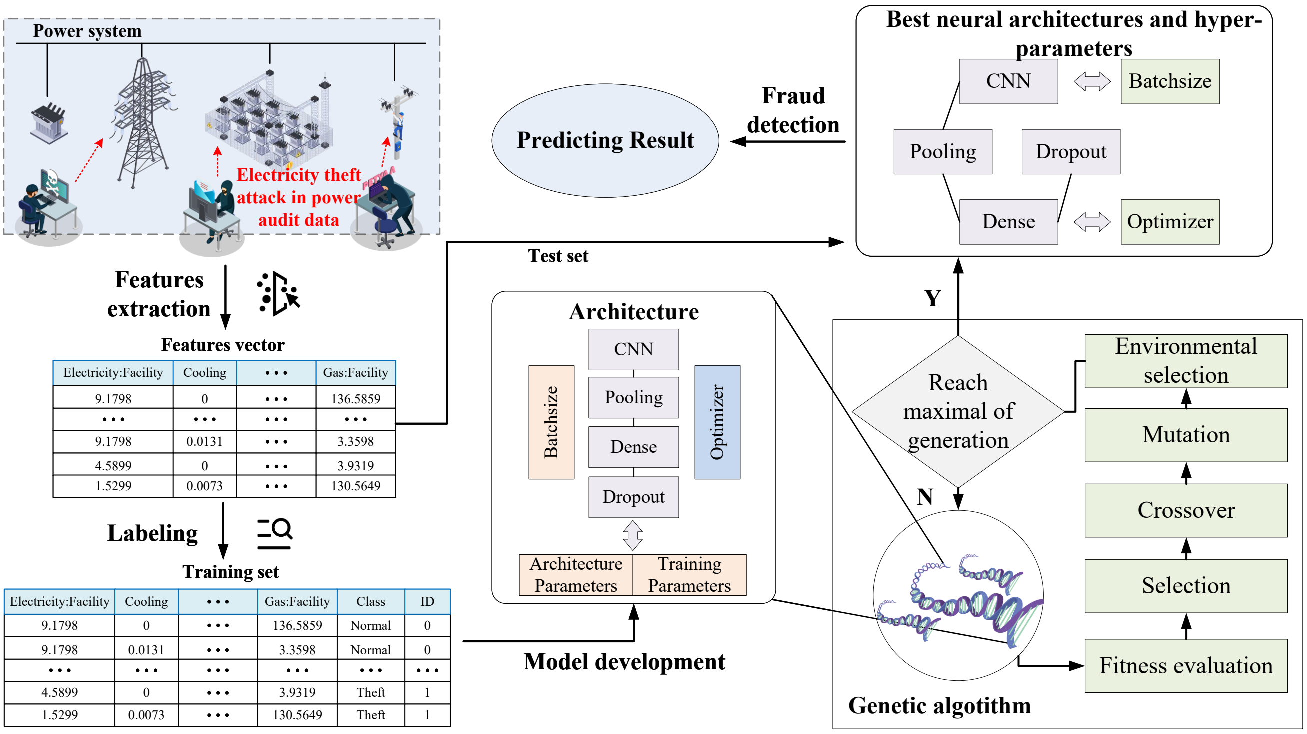
Step 5: Perform mutation operation. Select an individual and alter one of its gene values to produce a new individual.

Step 6: Termination condition check. If $gen > G_{max}$, terminate and output the optimal individual. Otherwise, set $gen = gen + 1$ and go to Step 2.

3. The Proposed NAS-FD Method

3.1. Algorithm Overview

Fig. 3 shows the overall framework of the proposed NAS-FD model for fraud detection in the power audit domain. It begins by processing outliers in a publicly available electricity theft attack dataset representing a power audit dataset, followed by its division into training and test sets, with subsequent data normalization applied to each. The partitioned and preprocessed dataset is then fed into the electricity theft attack detection optimization module. Initial parameters, including population size NP, maximum iterations G_{max}, training epochs $EP1$, test epochs $EP2$, crossover rate p_c, and mutation rate p_m, are set, and the decision variable ranges for optimization are determined. Subsequently, the GA-based model optimization process commences. An initial population P_0 of size NP is randomly generated, where each individual consists of a unique encoding representing the model neural architectures and hyper-parameters. The population iteration count is initialized to 1. Each individual undergoes a fitness evaluation, and the population is sorted in descending order based on fitness values. An environmental selection operation



Electricity:Facility	Cooling	• • •	Gas:Facility
9.1798	0	• • •	136.5859
• • •	• • •	• • •	• • •
9.1798	0.0131	• • •	3.3598
4.5899	0	• • •	3.9319
1.5299	0.0073	• • •	130.5649

Electricity:Facility	Cooling	• • •	Gas:Facility	Class	ID
9.1798	0	• • •	136.5859	Normal	0
9.1798	0.0131	• • •	3.3598	Normal	0
• • •	• • •	• • •	• • •	• • •	• • •
4.5899	0	• • •	3.9319	Theft	1
1.5299	0.0073	• • •	130.5649	Theft	1

Figure 3. The overall framework of the proposed NAS-FD model for fraud detection in power audit domain.

is then performed to generate a new population P_t, and the iteration count is incremented. During the predefined maximum iterations G_{max}, the evolutionary process repeats including a tournament selection mechanism that first generates a parent pool. Crossover and mutation operations are applied based on rates p_c and p_m to produce an offspring population. After the fitness evaluation, the offspring are merged with the current population. The combined population is sorted and undergoes environmental selection to choose superior individuals for the next generation, with the iteration count incremented. The algorithm terminates upon reaching G_{max}. Finally, the individual with the highest fitness in the last generation is selected as the best one. Its model neural architectures and hyper-parameters encoding are decoded. The decoded model, along with the preprocessed dataset, serves as input to the performance evaluation module. After comprehensive training and testing, the model that achieves superior performance indices is deployed for practical fraud detection tasks of power audit.

3.2. Population Initialization

Given the population size and the decision variable ranges, an initial population $P_i = \{p_1, \ldots, p_j, \ldots, p_{NP}\}$ containing NP individuals is randomly generated, where i denotes the current iteration count, and p_1, p_j, p_{NP} represent the first, j-th, and NP-th individuals in the population, respectively. The initial population is then returned. In this work, a real-valued vector encoding strategy is designed for the CNN neural architecture and hyper-parameters according to the decision variable ranges involved in the optimization. The model's neural architectures and hyper-parameters are represented as a real-number-based vector, with each individual's encoding in the population being a vector generated by this real-valued encoding strategy.

In the developed encoding strategy, an individual representing a model is a vector composed of real numbers and discrete numbers, specifically expressed as $p_i = \{Bs, Opt, Lr, Cnn, [Fil_1, Ker_1, Acf_1, Pol_1, Ptp_1, Pls_1, Std_1], \ldots, [Fil_l, Ker_l, Acf_l, Pol_l, Ptp_l, Pls_l, Std_l], Den, Dpr\}$.

The first part, $[Bs, Opt, Lr, Cnn]$, represents the training parameters and partial model control parameters of the CNN model, where Bs is the batch size during training, Opt denotes the type of neural network optimizer selected for CNN training. $Opt = 1$ is for Adam. $Opt = 2$ is for RMSprop. $Opt = 3$ is for SGD and $Opt = 4$ is for Adagrad. Lr is the learning rate and Cnn indicates the number of convolutional layers in the CNN model.

The second part, $[Fil_1, Ker_1, Acf_1, Pol_1, Ptp_1, Pls_1, Std_1], \ldots, [Fil_l, Ker_l, Acf_l, Pol_l, Ptp_l, Pls_l, Std_l]$, specifies the detailed parameters of the CNN modules, where l is the number of convolutional layers in a CNN module. Fil represents the number of filters in the convolutional layer, Ker is the kernel size, and Acf indicates the activation function type. $Acf = 1$ means ReLU and

$Acf = 2$ is Tanh. $Acf = 3$ is for Sigmoid and 4 for Softmax. Pol indicates whether a pooling layer is added after the convolutional layer, Ptp specifies the pooling type, Pls is the pooling window size, and Std is the pooling stride.

The third part, $[Den, Dpr]$, defines the parameters of the layers between the convolutional layers and the classification layer, where Den is the number of hidden neurons in the fully connected layer, and Dpr is the dropout rate in the dropout layer. In NAS-FD, all optimized variables are continuous, such as Bs ranging from 4 to 64. When discrete variables are involved, a binary encoding scheme is used to handle such cases, where a two-bit encoding can represent four variable values, a three-bit encoding eight values, and so forth.

3.3. Fitness Evaluation

Given a population P_t containing individuals to be evaluated, the number of training epochs $EP1$, and a preprocessed dataset $D = \{D_{\text{train}}, D_{\text{test}}\}$, a fitness evaluation is performed for each individual in the population. For each individual's fitness evaluation, the real-valued vector encoding is first decoded into a CNN model. Following the specified training configuration, this model is trained on the training set D_{train} for $EP1$ epochs. Finally, the test set D_{test} is used to evaluate the model and obtain its fitness value. This study selects the F1-score of the CNN model as the fitness value, where a higher F1-score indicates better classification performance. For an individual p in P_t, the specific formula of objective function is as follows:

$$f(p) = F1(p, EP, D) \tag{5}$$

where $F1$ denotes the calculation process for the F1-score of the model constructed from the individual p, EP represents the number of training epochs.

3.4. GA-Based Evolutionary Process

Given the crossover probability p_c and mutation probability p_m, crossover and mutation operations are applied to the current population P_t. First, a tournament selection is performed on the individuals in the population to obtain the parent population O_t. For each pair of adjacent individuals in O_t, a random number between 0 and 1 is generated. If this random number is less than p_c, crossover is applied to the pair.

During crossover, the first three parameters in the first part of both individuals' encodings undergo simulated binary crossover, while the parameter representing the number of convolutional layers in the CNN module remains unchanged. In the second part, due to potential differences in the number of convolutional layers between CNN modules, the smaller number of layers L_m is selected. Crossover is applied to the first L_m convolutional layers of both individuals, with the remaining layers unchanged. The convolutional layer crossover process is as follows: parameters representing filter size, number of kernels, activation function type, and pooling layer inclusion are first subjected to simulated binary crossover. Then, it is determined whether pooling layers are configured both before and after crossover. If both individuals have pooling layers configured after crossover, parameters representing pooling type, pool size, and stride undergo simulated binary crossover. Otherwise, a random number between 0 and 1 is generated. If less than 0.5, all pooling parameters are swapped between the two individuals. Parameters in the third part undergo simulated binary crossover.

After completing crossover operations on all individuals in the parent population, offspring individuals are obtained. Mutation operations are then applied to these offspring to yield the offspring population N_t. For each offspring individual, a random number between 0 and 1 is generated. If this number is less than p_m, mutation is applied. During mutation, parameters in the first part, i.e., Bs, Opt, Lr, undergo polynomial mutation, while Cnn remains unchanged. In the second part, parameters Fil, Ker, Acf, and Pol first undergo polynomial mutation. Then, it is checked whether the Pol value changes after mutation. If Pol remains 1, the remaining pooling parameters undergo polynomial mutation. If Pol changes from 0 to 1, random values are generated for the pooling type, pool size, and stride parameters. If Pol changes from 1 to 0, all pooling parameters are set to 0. If Pol remains 0, the mutation operation for the second part concludes, and mutation proceeds to the third part. In the third part, parameters representing the number of hidden neurons in the fully connected layer and the dropout rate both undergo polynomial mutation. In all simulated binary crossover and polynomial mutation operations, the control parameter η is set to 20.

4. Experimental Results and Analysis

All experiments are conducted on a workstation comprising an Intel Core i9-14900KF processor, 64 GB RAM, and an NVIDIA GeForce RTX 4060 Ti GPU. The software environment included Python 3.7.1, TensorFlow 2.3.4, NumPy 1.18.5, and Pandas 1.3.5. The parameter settings of NAS-FD are set as follows: $NP = 20$, $Itm = 10$, $EP1 = 50$, $EP2 = 90$, $p_c = 0.9$, and $p_m = 1/20$. $EP1$ is set to 50 because the model's trend becomes observable after 50 training epochs, and the training is terminated early to conserve computational time. $EP2$ is set to 90 instead of $EP1$, since it is used to evaluate the performance of the model represented by the final individual, whose performance metrics become stable after 90 training epochs.

4.1. Performance Indices

The performance of fraud detection models is evaluated by leveraging four widely adopted indices including accuracy, precision, recall, and F1-score. These indices are defined as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

$$Recall = \frac{TP}{TP + FN} \tag{7}$$

$$Precision = \frac{TP}{TP + FP} \tag{8}$$

$$F_1\text{-}score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{9}$$

where TP, TN, FP, and FN signify true positives, true negatives, false positives, and false negatives. TP characterizes accurately detected frauds instances, and TN characterizes correctly classified normal samples. Conversely, FP signifies normal samples incorrectly flagged as fraud, and FN signifies frauds that are undetected and misclassified as normal.

4.2. Datasets and Preprocessing

To validate the effectiveness of the proposed NAS-FD method, the TDD2022 dataset [8], which is commonly used in the power auditing domain, is employed. A detailed description is given as follows. The TDD2022 dataset is a multi-class theft detection dataset specifically designed for benchmarking in the smart grid domain. It is generated by applying an effective theft generator model to real-world, publicly available energy consumption data from the Open Energy Data Initiative platform. This methodology transforms standard consumption data into a valuable resource for detecting anomalous patterns. In this paper, the multi-class dataset is converted into a binary classification dataset for model validation. In addition, all features are normalized using the z-score method, which is defined as follows:

$$x_i' = \frac{x_i - \mu}{\sigma} \tag{10}$$

where x_i and x_i' denote the original and normalized values of the i-th feature, respectively. μ is the mean value. σ is the standard deviation. For the TDD2022 dataset, a classical split ratio of 0.8 is adopted for the training set, while the remaining data are used for testing, so that the model performance can be effectively evaluated during the fitness assessment.

4.3. Evolutionary Trajectories

Fig. 4 presents the fraud detection performance of NAS-FD during the power audit process. The objective function is the F1-score, where a higher value indicates better model performance. As shown in Fig. 4, the initial F1-score is below 0.87. As evolution progresses, the F1-score exceeds 0.91, demonstrating that the NAS technique automatically achieves superior performance without relying on expert knowledge. Furthermore, the algorithm begins to converge from the sixth generation, indicating that the best solution has been obtained. The solution from the final generation is decoded, yielding: [62, 1, 0.0008, 5], [79, 3, 1, 0, 0, 0, 0], [163, 3, 1, 0, 0, 0, 0], [228, 3, 1, 0, 0, 0, 0], [403, 2, 3, 1, 1, 2, 3], [464, 2, 1, 1, 1, 3, 3], [253, 0.3687]. The corresponding CNN architecture is shown in Fig. 5. Once the optimized CNN is obtained, it can be deployed into the system. Notably, the process of searching for the neural architectures and hyper-parameters of CNN is conducted offline, but the deployed model enables online detection.

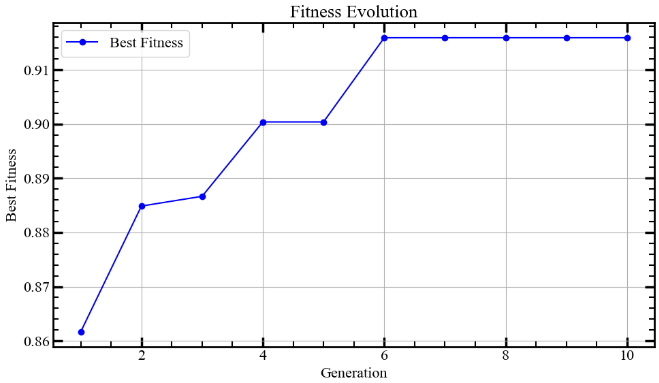


Figure 4. The evolutionary trajectory of the NAS-FD for power audit data.

Table 1. Performance Comparison of NAS-FD, CNN, LSTM on TTD2022 Dataset

Metrics	NAS-FD	CNN	LSTM
Accuracy	**0.9193**	0.9046	0.8582
Precision	**0.9214**	0.9085	0.8594
Recall	**0.9193**	0.9046	0.8582
F1-Score	**0.9185**	0.9033	0.8566

4.4. Comparison with Existing Manually Designed Models

This subsection compares NAS-FD with manually designed CNN and LSTM models. The manually designed CNN is selected because NAS-FD automates the design of CNN neural architectures and hyper-parameters, thereby demonstrating the effectiveness of the NAS technology. The manually designed LSTM

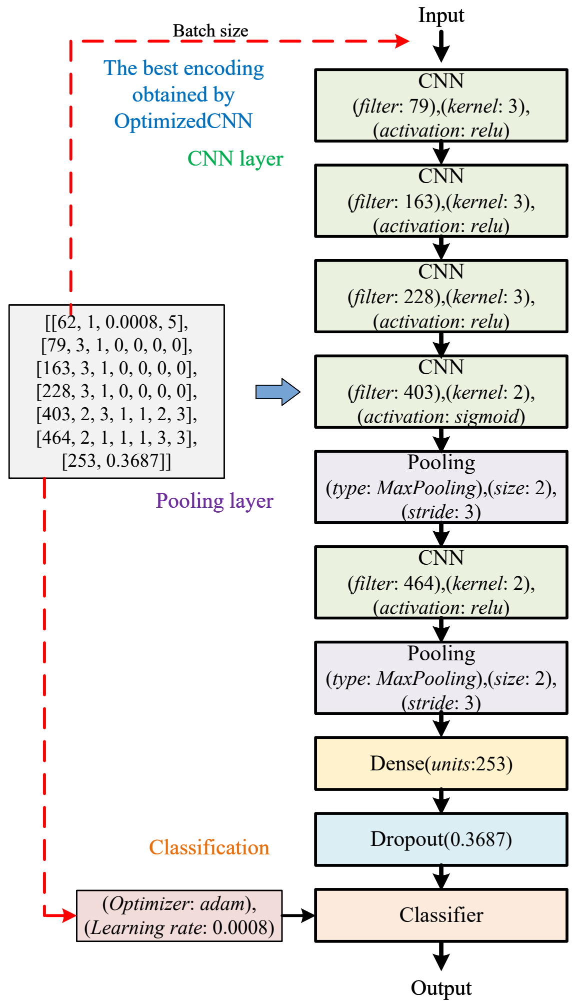
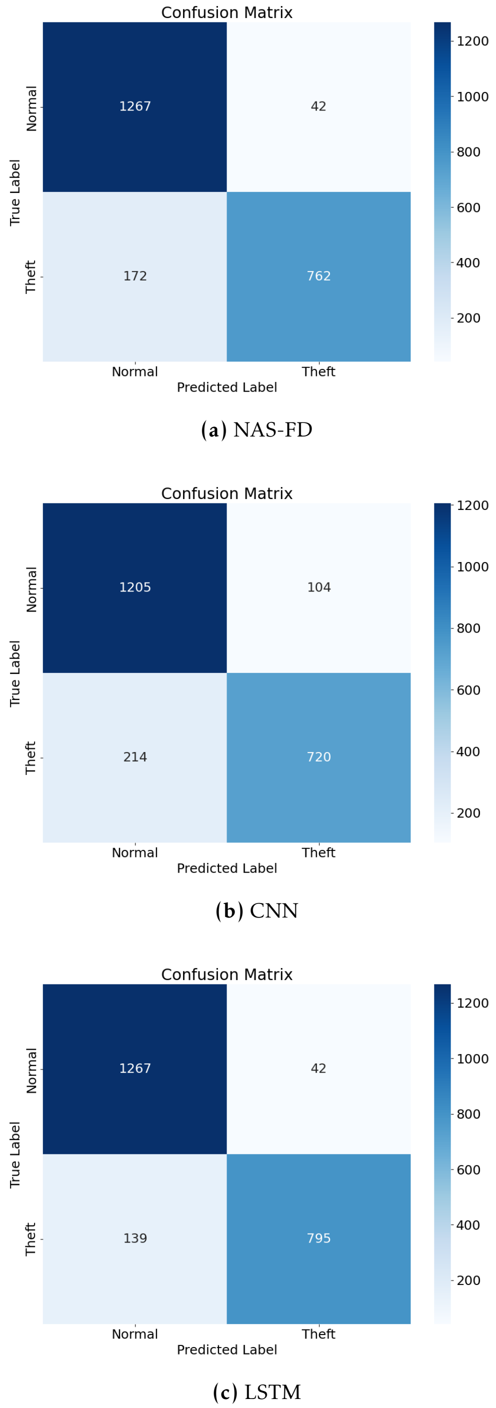

Input
Batch size
The best encoding obtained by OptimizedCNN
CNN layer
CNN (filter: 79),(kernel: 3), (activation: relu)
CNN (filter: 163),(kernel: 3), (activation: relu)
CNN (filter: 228),(kernel: 3), (activation: relu)
[[62, 1, 0.0008, 5], [79, 3, 1, 0, 0, 0, 0], [163, 3, 1, 0, 0, 0, 0], [228, 3, 1, 0, 0, 0, 0], [403, 2, 3, 1, 1, 2, 3], [464, 2, 1, 1, 1, 3, 3], [253, 0.3687]]
CNN (filter: 403),(kernel: 2), (activation: sigmoid)
Pooling layer
Pooling (type: MaxPooling),(size: 2), (stride: 3)
CNN (filter: 464),(kernel: 2), (activation: relu)
Pooling (type: MaxPooling),(size: 2), (stride: 3)
Dense(units:253)
Classification
Dropout(0.3687)
(Optimizer: adam), (Learning rate: 0.0008)
Classifier
Output

Figure 5. The optimized CNN model achieved by the NAS-FD method.

is included to illustrate that CNN is more suitable than LSTM for fraud detection in power audit data. The architectures and hyperparameters of the two compared models were set to their best-performing values obtained through manual tuning, ensuring a fair and reliable comparison under the same experimental conditions.

Table 1 presents the performance comparison of NAS-FD, CNN, and LSTM on the TTD2022 dataset. Fig. 6 gives the confusion matrix obtained by NAS-FD, CNN, and LSTM. The results show that NAS-FD achieves the best performance with values of 0.9193, 0.9214, 0.9193, and 0.9185 for accuracy, precision, recall, and F1-score, respectively. Significant improvements are observed compared to CNN and LSTM, particularly relative to LSTM's results of 0.8582, 0.8594, 0.8582, and 0.8566. These results demonstrate the effectiveness of NAS technology for fraud detection in power audit data. Furthermore, the comparison reveals that both NAS-FD and CNN outperform LSTM, indicating that CNN-based models exhibit greater potential than LSTM. Notably, the neural architectures and hyper-parameters of NAS-FD are automatically designed without manual trial-and-error, representing an additional advantage over manually designed CNN and LSTM models.



(a) NAS-FD

(b) CNN

(c) LSTM

Figure 6. The confusion matrix obtained by NAS-FD, CNN, and LSTM.

5. Conclusion

In this paper, we have proposed a NAS-based fraud detection method termed NAS-FD for power audit data. In NAS-FD, the automated design of CNN, including both neural architectures and hyper-parameters, is formulated as a single-objective optimization problem aimed at maximizing the F1-score. Furthermore, GA is employed as the primary optimization framework. Within this framework, an effective encoding strategy along with corresponding crossover and mutation operations is developed to represent and evolve the neural architectures and hyper-parameters of CNN, thereby overcoming the limitations of manual trial-and-error. To validate the effectiveness of NAS-FD, comparative experiments are conducted on an electricity theft dataset, a common challenge in power auditing. Compared with manually designed CNN and LSTM models, the experimental results demonstrate that NAS-FD achieves superior detection performance in terms of four common performance indices including accuracy, precision, recall, and F1-score.

Future research will consider fraud detection model design from a multi-objective perspective by incorporating model complexity as one objective. Additionally, unknown fraud scenarios will be investigated using unsupervised learning approaches for power audit data analysis. The proposed method could also be extended in the future to multi-objective privacy-preserving tasks [23] and generic neural networks [24].

References

[1] Huang, H., Vincent Poor, H., Davis, K.R., Overbye, T.J., Layton, A., Goulart, A.E. and Zonouz, S. (2024) Toward resilient modern power systems: From single-domain to cross-domain resilience enhancement. *Proceedings of the IEEE* **112**(4): 365–398. doi:10.1109/JPROC.2024.3405709.

[2] Xu, D., Niu, W., Li, Q., Li, H. and Cheng, L. (2024) Enhancing power marketing audit through iot and multi-sensor information fusion: A substation scenario analysis. *Computers and Electrical Engineering* **118**: 109312.

[3] Wang, C., Yu, X., Tan, G. and Xiao, L. (2024) Utilization of blockchain technology in the data audit system of power grid engineering. *Procedia Computer Science* **243**: 172–179.

[4] Zhou, W., Li, B., Xiao, H., Xiao, H., Wang, W., Zheng, Y. and Su, S. (2024) Electricity theft detection of residential users with correlation of water and electricity usage. *IEEE Transactions on Industrial Informatics* **20**(4): 5339–5351. doi:10.1109/TII.2023.3332954.

[5] Mishra, A.K. and Das, B. (2025) A novel density based clustering approach for electricity theft detection. *IEEE Transactions on Industry Applications* **61**(4): 5537–5548. doi:10.1109/TIA.2025.3544167.

[6] Yan, Z. and Wen, H. (2022) Performance analysis of electricity theft detection for the smart grid: An overview. *IEEE Transactions on Instrumentation and Measurement* **71**: 1–28. doi:10.1109/TIM.2021.3127649.

[7] Omoteso, K. (2012) The application of artificial intelligence in auditing: Looking back to the future. *Expert Systems with Applications* **39**(9): 8490–8495.

[8] Zidi, S., Mihoub, A., Qaisar, S.M., Krichen, M. and Al-Haija, Q.A. (2023) Theft detection dataset for benchmarking and machine learning based classification in a smart grid environment. *Journal of King Saud University-Computer and Information Sciences* **35**(1): 13–25.

[9] Zhu, S., Xue, Z. and Li, Y. (2024) Electricity theft detection in smart grids based on omni-scale cnn and autoxgb. *IEEE Access* **12**: 15477–15492. doi:10.1109/ACCESS.2024.3358683.

[10] Xia, X., Lin, J., Jia, Q., Wang, X., Ma, C., Cui, J. and Liang, W. (2023) ETD-ConvLSTM: A deep learning approach for electricity theft detection in smart grids. *IEEE Transactions on Information Forensics and Security* **18**: 2553–2568. doi:10.1109/TIFS.2023.3265884.

[11] Liu, Y., Sun, Y., Xue, B., Zhang, M., Yen, G.G. and Tan, K.C. (2023) A survey on evolutionary neural architecture search. *IEEE Transactions on Neural Networks and Learning Systems* **34**(2): 550–570. doi:10.1109/TNNLS.2021.3100554.

[12] Liang, J., Liu, G., Bi, Y., Yu, M., Liu, M. and Jin, Y. (2025) Evolutionary neural architecture search for remote sensing image classification. *IEEE Transactions on Neural Networks and Learning Systems* **36**(10): 17886–17900. doi:10.1109/TNNLS.2025.3579517.

[13] Real, E., Moore, S., Selle, A., Saxena, S., Suematsu, Y.L., Tan, J., Le, Q.V. *et al.* (2017) Large-scale evolution of image classifiers. In *International Conference on Machine Learning* (PMLR): 2902–2911.

[14] Sun, Y., Yen, G.G. and Yi, Z. (2019) Evolving unsupervised deep neural networks for learning meaningful representations. *IEEE Transactions on Evolutionary Computation* **23**(1): 89–103. doi:10.1109/TEVC.2018.2808689.

[15] Ho, K., Gilbert, A., Jin, H. and Collomosse, J. (2021) Neural architecture search for deep image prior. *Computers & Graphics* **98**: 188–196.

[16] Zeng, G.Q., Yang, Y.W., Lu, K.D., Geng, G.G. and Weng, J. (2025) Evolutionary adversarial autoencoder for unsupervised anomaly detection of industrial internet of things. *IEEE Transactions on Reliability* **74**(3): 3454–3468. doi:10.1109/TR.2025.3528256.

[17] Lu, K.D., Yang, Y.W., Zeng, G.Q., Peng, C., Geng, G.G. and Weng, J. (2025) BPSO-AHDL-IDS: Binary particle swarm optimization-based automated hybrid deep learning model for intrusion detection of internet of things. *IEEE Transactions on Automation Science and Engineering* **22**: 15859–15877. doi:10.1109/TASE.2025.3572510.

[18] Dong, L., Li, Q., Wu, K., Fei, K., Liu, C., Wang, N., Yang, J. *et al.* (2020) Nontechnical loss detection of electricity based on neural architecture search in distribution power networks. In *2020 International Conference on Smart Grid and Clean Energy Technologies (ICSGCE)*: 143–148. doi:10.1109/ICSGCE49177.2020.9275605.

[19] Fei, K., Li, Q. and Zhu, C. (2022) Non-technical losses detection using missing values' pattern and neural

architecture search. *International Journal of Electrical Power and Energy Systems* **134**: 107410.

[20] Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M. and Inman, D.J. (2021) 1d convolutional neural networks and applications: A survey. *Mechanical Systems and Signal Processing* **151**: 107398.

[21] Alzubaidi, L., Zhang, J., Humaidi, A.J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J. *et al.* (2021) Review of deep learning: concepts, cnn architectures, challenges, applications, future directions. *Journal of Big Data* **8**(1): 53.

[22] Katoch, S., Chauhan, S.S. and Kumar, V. (2021) A review on genetic algorithm: past, present, and future. *Multimedia Tools and Applications* **80**(5): 8091–8126.

[23] Ge, Y.F., Wang, H., Bertino, E., Cao, J. and Zhang, Y. (2025) Multiobjective privacy-preserving task assignment in spatial crowdsourcing. *IEEE Transactions on Cybernetics* **55**(8): 3584–3597. doi:10.1109/TCYB.2025.3573292.

[24] Tawhid, M.N.A., Siuly, S., Wang, K. and Wang, H. (2024) Genet: A generic neural network for detecting various neurological disorders from eeg. *IEEE Transactions on Cognitive and Developmental Systems* **16**(5): 1829–1842. doi:10.1109/TCDS.2024.3386364.