Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 6650784, 11 pages
https://doi.org/10.1155/2021/6650784

Research Article

FPETD: Fault-Tolerant and Privacy-Preserving Electricity Theft Detection

Siliang Dong, Zhixin Zeng, and Yining Liu

School of Computer and Information Security, Guilin University of Electronic Technology, China

Correspondence should be addressed to Yining Liu; lyn7311@sina.com

Received 18 November 2020; Revised 8 December 2020; Accepted 26 May 2021; Published 10 June 2021

Academic Editor: Nathalie Mitton

Electricity theft occurs from time to time in the smart grid, which can cause great losses to the power supplier, so it is necessary to prevent the occurrence of electricity theft. Using machine learning as an electricity theft detection tool can quickly lock participants suspected of electricity theft; however, directly publishing user data to the detector for machine learning-based detection may expose user privacy. In this paper, we propose a real-time fault-tolerant and privacy-preserving electricity theft detection (FPETD) scheme that combines n-source anonymity and a convolutional neural network (CNN). In our scheme, we designed a fault-tolerant raw data collection protocol to collect electricity data and cut off the correspondence between users and their data, thereby ensuring the fault tolerance and data privacy during the electricity theft detection process. Experiments have proven that our dimensionality reduction method makes our model have an accuracy rate of 92.86% for detecting electricity theft, which is much better than others.

1. Introduction

Electricity theft is widespread in the smart grid [1]; illegal users may be trying to reduce their bills by stealing electricity. Electricity theft will cause great economic losses to the power company and potential safety hazards such as fire and electric shock [2, 3]. Therefore, it is vital to take measures to detect electricity theft behaviors in real time. The Internet of Things (IoT) is popular nowadays; in the smart grid, users' electricity consumption data are collected from smart meters to the data center in real time, which can be published or outsourced for data analysis to identify the theft users [4, 5].

Many works have been conducted to solve the electricity theft detection problem in the smart grid. The existing electricity theft detection methods are mainly divided into three categories: state estimation, game theory, and machine learning. Among them, the machine learning-based methods are now widely used in electricity theft detection since they are more efficient and accurate.

However, most of the existing machine learning-based schemes for electricity theft detection consider the detection server to be credible [6–9]. For example, in [8], the authors use decision tree and support vector machine (SVM) to conduct electricity theft detection. The detector directly obtains the raw data corresponding to the user, which will leak the user's privacy. In [6], although the data center can detect the number of stealers, the detector can decrypt users' encrypted data and directly obtain users' raw data. In reality, the detection server is often untrusted. Directly publishing users' data to the detection server may reveal user privacy [10, 11]. Now is the era of big data, and data contains sensitive information of people. For example, an attacker may analyze the user's electricity consumption data to find out whether the user is at home or not on a certain day, which may cause security issues such as burglary, so it is important to protect user data privacy.

To protect user privacy, it is necessary to perform privacy processing on user data before publishing it to the detector server [12]. Common data privacy processing technologies such as data aggregation [13, 14] have well achieved privacy-preserving data collection, but they cannot be combined with the actual application of electricity theft detection [15, 16]. For example, in [13], each user encrypts his/her own electricity consumption data into a ciphertext using a lifted

ElGamal homomorphic cryptosystem before sending it to the aggregator [17]. The aggregator aggregates ciphertexts from all users and sends the aggregated data to the operating center [18]. In this way, the operating center obtains the sum of all. The aggregated data reflects the overall electricity consumption level; the data published facilitates the electricity distribution decision. However, the data aggregation scheme cannot achieve the accurate electricity theft detection in a specific area because the sum data may not retain the features of the original single user's raw data [19]. The electricity theft detection needs to extract the features of a single user for analysis. But the aggregation result may mask these features; it is not applicable for electricity theft detection.

Compared with the maximum value, average value, and sum value obtained by data aggregation, the raw data obtained by the n-source anonymity method often has greater use-value. The n-source anonymity is a data privacy processing method which was first proposed by Zhang et al. [20]; cryptographic tools are used to guarantee the rawness and unlinkability of the data. In Liu et al.'s scheme [21], "Shuffle" is introduced to allocate the participants' slots, to achieve n-source anonymity without a trusted third party. After that, Chen et al. [22] improve Liu et al.'s scheme [21] to reduce storage efficiency and make the scheme more lightweight. The anonymized raw data collected by the n-source anonymity method can realize the detection of electricity theft under the premise of protecting user privacy. However, the existing n-source anonymity methods [20–22] do not take into account the fault tolerance issues that may occur during the data collection stage in real data application scenarios. In the data collection stage of a real electricity theft detection scenario, if a device fails, it is likely that the entire detection system cannot work normally; therefore, fault tolerance is important. In this paper, a fault-tolerant and privacy-preserving electricity theft detection (FPETD) scheme is proposed, and three contributions are achieved as follows.

(1) We perform privacy processing on the users' electricity consumption data by n-source anonymity before it is published, to complete real-time electricity theft detection without the need of a trusted third party while ensuring user privacy

(2) We propose a fault-tolerant n-source anonymity data collection scheme, so that users' electricity consumption data can still be collected privately in the event some smart meters fail, thereby ensuring that electricity theft detection can still be performed normally in the case of device failure

(3) Sufficient experiments prove that the data normalization and dimensionality reduction preprocessing we do on the dataset can speed up the model training speed and improve the detection accuracy. Our preprocessing of the dataset makes our CNN model perform better than other existing methods

The rest of the paper is organized as follows. Preliminaries are introduced in Section 2. System model and security threats are discussed in Section 3. The proposed FPETD scheme is presented in Section 4. Privacy analysis is introduced in Section 5. Experiments and analysis are described in Section 6. Finally, the paper is concluded in Section 7.

2. Preliminaries

2.1. n-Source Anonymity. n-source anonymity is a raw data collection method that cannot trace the source of data. The steps of n-source anonymity in Liu et al.'s scheme [21] are as follows (Figure 1 shows a process of 4-source anonymity data collection):

(1) Each participant p_i $(i \in [1, n])$ obtains a slot which is only known by himself/herself through the slot generation phase (for more details, the reader can refer [21])

(2) Each participant p_i $(i \in [1, n])$ uses the session keys to construct the masking data e_i^j $(i \in [1, n], j \in [1, n])$ and adds raw data m_i to the slot(i)-th slot. Then, each p_i obtains a ciphertext:

$$c_i = e_i^1 | e_i^2 | \cdots | e_i^{\text{slot}(i)} \oplus m_i | \cdots | e_i^n. \tag{1}$$

(3) The data collector executes the XOR operation on the ciphertexts from the participants and obtains the raw data finally

2.2. z-Score Standardization. z-score standardization is often used to map all features of the data to the same scale to avoid part of features of the data from forming a leading role due to numerical magnitude differences. The equation of the z-score standardization performs as below:

$$x_i = \frac{\bar{x}_i - \hat{x}}{\sigma^2}, \quad \forall i, \tag{2}$$

where $\hat{x}$ and σ represent the mean and standard deviation of the training samples. The $\bar{x}_i \in R^n, i = 1, \cdots, N$, represent a training set of points. The $x_i \in R^n, i = 1, \cdots, N$, represent the normalized training samples.

2.3. Principal Component Analysis. Principal component analysis (PCA) is often used to reduce the dimensionality of a dataset while maintaining the features that contribute the largest variance in the dataset. PCA is the most commonly used linear dimensionality reduction method [23]. Dimensionality reduction plays an essential role in machine learning, especially when the dataset has more than a thousand features. In addition to making feature processing easier, the algorithm can also improve the results of the classifier and speed up the training of the classifier. In dimensionality reduction, the information measurement indicator used by PCA is the sample variance, also known as the explainable variance; the greater the variance, the

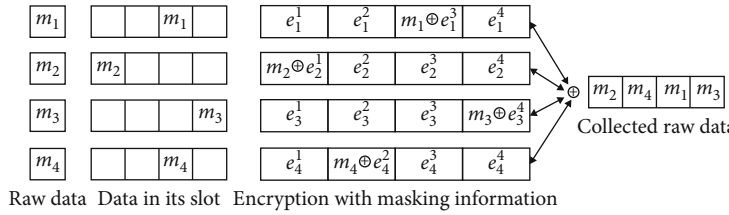

Figure 1: A process of 4-source anonymity.

more information the feature carries. The feature variance equation is as below:

$$v_{ar} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - x\wedge)^2, \tag{3}$$

where v_{ar} represents the variance of a feature, n represents the number of samples, x_i represents the value of each sample in a feature, and $\hat{x}$ represents the mean of this list of samples.

2.4. Convolutional Neural Network. Convolutional neural network (CNN) is a common model in the field of deep learning, which is often used to deal with large-scale image problems [24]. CNN includes convolutional layers, pooling layers, and full connection layers [25]. The common CNN model structure is shown in Figure 2.

The convolutional layer includes multiple convolution kernels. These convolution kernels slide on the input matrix, that is, do dot multiplication with the pixels of the matrix to obtain a new matrix, which is the feature map. These feature maps are used as the next layer entry. The ReLU activation function is used to realize the nonlinear classification of neural networks. The equation of ReLU [26] performs as follows:

$$f(x) = \begin{cases} 0, & x < 0, \\ x, & x \geq 0. \end{cases} \tag{4}$$

Pooling layer: maximum pooling selects a maximum number in the sliding window as the result. The role of the pooling layer is to reduce the dimension of the feature map and reduce the amount of calculation.

Full connection layer: each node of this layer is connected to all nodes of the previous layer, which combines the features obtained from all previous layers and outputs them to the softmax function [27] for classification.

The softmax function is used to obtain the probability of the categories, which compresses the elements in the K -dimensional vector output by the full connection layer to the range of (0,1], and the result of their addition is 1,

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_k^K e_{z_k}} \quad \text{for } j = 1, \cdots, K. \tag{5}$$

The final predicted categories are compared with the real categories, and through backpropagation [28], the parameters in the convolutional neural network are updated again and again in the iteration of backpropagation, approaching our real parameters infinitely. To train the CNN, we use cross entropy as a loss function; the equation of cross entropy for the distributions u and v over a given discrete set performs as below:

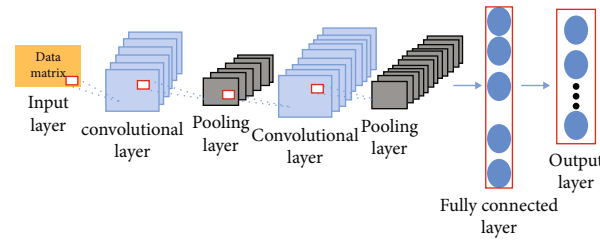

Figure 2: Classic CNN structure diagram.

$$H(u, v) = -\sum_x u(x) \log v(x). \tag{6}$$

3. System Model and Security Threats

3.1. System Model. As shown in Figure 3, the system model includes participants p_i ($i \in [1, n]$), fog nodes (FN), the cloud server (CS), and the data consumer.

(1) Data consumer serves as a detection server in the system

(2) Each participant equipped with a smart meter collects the real-time electricity consumption data and sends the data to FN

(3) FN is a server located near the participants for processing data, which performs fog computing and reduces the computational burden of CS [29]. FN communicates with the participants and forwards participants' real-time electricity consumption data to CS according to a set of protocols [30, 31]

(4) CS stores the electricity consumption data sent by the FN. Data collected by CS can be outsourced to a data consumer for further processing, such as electricity theft detection

3.2. Security Threats. The smart grid suffers from a variety of security issues, such as the data injection attack, the denial of service attack, and some other physical threats [32]. Security goals require that the data is only shared between the two

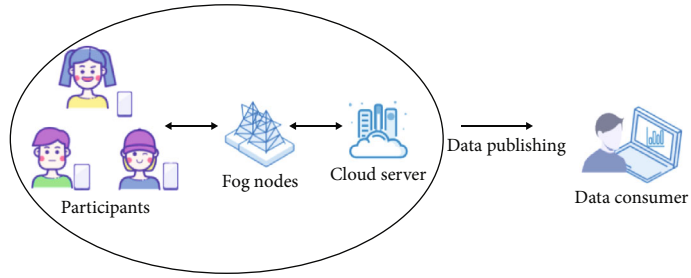

Figure 3: System model.

authenticated entities [33]. However, these security goals are not enough in the real environment; the privacy issues are also very important. For example, the data consumers are often untrusted in reality; they often leak user data for profit. The data privacy processing system in the ellipse in Figure 3 will process the data privacy to achieve a balance between participants' data privacy and the needs of the data consumer under the premise that the data consumer is untrusted. The privacy goals require that the data source cannot be known by others, and the data are not totally shared between two entities. In the proposed fault-tolerant and privacy-preserving electricity theft detection (FPETD) scheme, we assume that there exists a secure communication channel between the authenticated entities, and hence, we only consider privacy issues in our system model. Therefore, the security threats of the proposed FPETD scheme are as follows:

(1) The data consumer is untrusted which is curious for users' privacy

(2) CS and FN are honest-but-curious. That is to say, CS and FN will not tamper with the data but will infer the source of the data

(3) Participants are also honest-but-curious. Each participant will honestly follow the protocol but try to snoop on others' data

4. Our Proposed FPETD Scheme

This section presents the detailed content of the proposed FPETD scheme; it mainly includes the data collection phase with fault tolerance, method for faulty participants to reconnect, example for fault tolerance and reconnection, and detector design and training phase. In the data collection phase, the fault tolerance is introduced to ensure that the data collection process can still be completed normally in the case of device failure, and the method for faulty participants to reconnect is introduced to ensure the faulty participants can reconnect normally. In the detector design and train phase, we propose an efficient method to train a CNN-based detection model and use the trained model to detect the electricity theft. The overall system flow chart is shown in Figure 4.

4.1. Data Collection Phase with Fault Tolerance. Here, we mainly introduce data collection when a failure occurs. In the real situation of data collection, even if the network situation is good, some participants will occasionally be offline. The section describes the data collection process when some participants fail to work; the cloud server can still collect data of the normal participants.

Session key sharing: each participant $p_i\ (i \in [1, n])$ selects $(1 \le \beta \le n-1)$ key-shared partners in the group and shares a session key $k_{ij}\ (i, j \in [1, n], i \ne j)$ with the selected partners p_j. p_i stores all the session keys $\{k_{i1}, k_{i2}, \cdots, k_{i\beta}\}\ (1 \le \beta \le n-1)$ in the local.

CS sends data collection requests to the FN. FN initiates a task request. Assuming that only $b\ (1 < b \le n-1)$ participants respond, FN notifies the normal participants p_i to check if its key-shared partners are functional; if not, p_i removes the corresponding session keys from its list. After that, assuming p_i possesses $\alpha\ (1 \le \alpha \le b-1)$ session keys, where α is the number of the session keys of the normal participants p_i,, then p_i performs:

(1) Each p_i reconstructs new masking data $e_i^j\ (j \in [1, n])$ with time t (time t is the timestamp in every time period) such as

$$\begin{cases} e_i^1 = h(k_{i1}|t|1) \oplus h(k_{i2}|t|1) \cdots \oplus h(k_{i\alpha}|t|1), \\ e_i^2 = h(k_{i1}|t|2) \oplus h(k_{i2}|t|2) \cdots \oplus h(k_{i\alpha}|t|2), \\ \cdots \\ e_i^n = h(k_{i1}|t|n) \oplus h(k_{i2}|t|n) \cdots \oplus h(k_{i\alpha}|t|n). \end{cases} \tag{7}$$

(2) Each p_i adds his/her raw data m_i to the slot(i)-th slot, and the ciphertext c_i is reconstructed as follows:

$$c_i = e_i^1 | e_i^2 | \cdots | e_i^{\text{slot}(i)} \oplus m_i | \cdots | e_i^n. \tag{8}$$

(3) FN eventually receives $c_i\ (i = 1, 2, \cdots, b)$ from the b participants. FN executes the XOR operation to all the $c_i\ (i = 1, 2, \cdots, b)$ and obtains a collected data list $\text{ML} = \{m_{\pi_{n(1)}}, m_{\pi_{n(2)}}, \cdots, m_{\pi_{n(n)}}\}$. It is worth mentioning

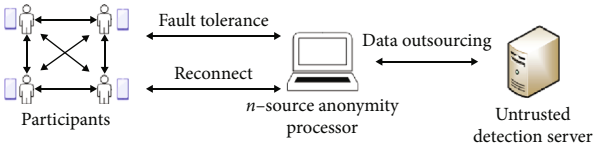

FIGURE 4: System flow chart.

that the collected messages of the faulty participants are 0. Then, FN sends the collected electricity consumption data list ML to CS

4.2. Method for Faulty Participants to Reconnect. Assuming all the faulty participants want to reconnect to the network in the next data collection task, the faulty participants apply for new session keys with others in the group according to Session Key Sharing in Section 4.1. Then, p_i possesses $(1 \le \beta \le n-1)$ session keys; p_i performs

(1) p_i reconstructs new masking data $e_i^j\ (j \in [1, n])$ with time t as

$$\begin{cases} e_i^1 = h(k_{i1}|t|1) \oplus h(k_{i2}|t|1) \cdots \oplus h\left(k_{i\beta}|t|1\right), \\ e_i^2 = h(k_{i1}|t|2) \oplus h(k_{i2}|t|2) \cdots \oplus h\left(k_{i\beta}|t|2\right), \\ \cdots \\ e_i^n = h(k_{i1}|t|n) \oplus h(k_{i2}|t|n) \cdots \oplus h\left(k_{i\beta}|t|n\right). \end{cases} \tag{9}$$

(2) p_i adds m_i to the slot(i)-th slot, and c_i is reconstructed as follows:

$$c_i = e_i^1|e_i^2| \cdots |e_i^{\text{slot}(i)} \oplus m_i| \cdots |e_i^n. \tag{10}$$

(3) FN eventually receives $c_i\ (i = 1, 2, \cdots, n)$ from the n participants. FN executes the XOR operation to all the $c_i\ (i = 1, 2, \cdots, n)$ and obtains a collected data list $\text{ML} = \{m_{\pi_{n(1)}}, m_{\pi_{n(2)}}, \cdots, m_{\pi_{n(n)}}\}$. Then, FN sends the collected electricity consumption data list ML to CS

4.3. An Example of Fault Tolerance and Reconnection. We use 4 participants as an example to describe the data collection with fault tolerance and the method for faulty devices to reconnect. p_1 shared $k_{12}, k_{13},$ and k_{14} with p_2, p_3, and p_4, respectively, while p_2 shared k_{23} with p_3, and after the slot generation, their slots are 3, 1, 4, and 2, respectively.

(1) Assuming that every time the data collection is performed on one floor in a building, there are 4 participants on each floor. CS sends task collection requests to the FN. FN initiates a task request. Assuming that p_1, p_3, and p_4 respond, and p_2 does not respond, it is considered that p_2 is faulty. Data on the corresponding slot of the faulty p_2 is 0 after the XOR operation. In order to prevent the slot of the faulty participant from being exposed, FN randomly selects a participant from normal participants and informs him/her to submit data 0, thereby effectively preventing the leakage of the slot of the faulty participant. Assume that the FN randomly selects normal p_3 and informs him/her to fill in the data with 0

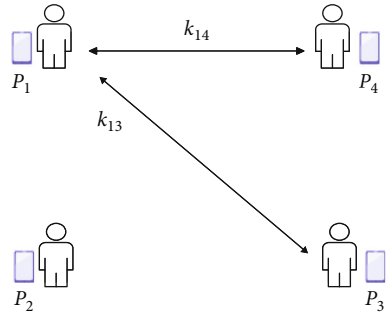

FIGURE 5: Session key in fault state.

(2) FN notifies the normal participants p_1, p_3, and p_4 to check if its key-shared partners are functional; those who have the session keys with the faulty p_2 will delete the session keys with p_2, that is, delete k_{12} and k_{23}, as shown in Figure 5

(3) Then, p_i reconstructs the masking data

p_1 reconstructs new masking data.

$$\begin{aligned} e_1^1 &= h(k_{13} \mid 1) \oplus h(k_{14} \mid 1), \\ e_1^2 &= h(k_{13} \mid 2) \oplus h(k_{14} \mid 2), \\ e_1^3 &= h(k_{13} \mid 3) \oplus h(k_{14} \mid 3), \\ e_1^4 &= h(k_{13} \mid 4) \oplus h(k_{14} \mid 4). \end{aligned} \tag{11}$$

Because the slot of p_1 is 3, the data that p_1 fills in the third slot is $e_1^3 \oplus m_1$, and finally, the masking data is filled into each slot in order and the ciphertext $c_1 = e_1^1|e_1^2|e_1^3 \oplus m_3|e_1^4$ of the data collection phase of p_1 is obtained. p_2 is faulty at this time and does not do anything.

The ciphertext of the data collection phase of p_3 is $c_3 = e_3^1 \mid e_3^2 \mid e_3^3 \mid e_3^4 \oplus m_3$.At this time, p_3 is selected by the FN, and the data is filled with 0 for disguise to protect the slot of the faulty participant, that is, $m_3 = 0$.

$$\begin{aligned} e_3^1 &= h(k_{13} \mid 1), \\ e_3^2 &= h(k_{13} \mid 2), \\ e_3^3 &= h(k_{13} \mid 3), \\ e_3^4 &= h(k_{13} \mid 4). \end{aligned} \tag{12}$$

$c_4 = e_4^1 \mid e_4^2 \oplus m_4 \mid e_4^3 \mid e_4^4$ is the ciphertext of the data collection phase of p_4.

$$\begin{aligned} e_4^1 &= h(k_{14} \mid 1), \\ e_4^2 &= h(k_{14} \mid 2), \\ e_4^3 &= h(k_{14} \mid 3), \\ e_4^4 &= h(k_{14} \mid 4). \end{aligned} \tag{13}$$

In the end, each participant publishes his/her data collection phase ciphertext to FN, and FN performs the XOR operation to all the ciphertexts, that is, $(c_1 \oplus c_3 \oplus c_4)$, to obtain $0|m_4|m_1|0$.In the next data collection task, the faulty p_2 applies to reconnect. At this time, the faulty p_2 applies for the new session keys, as shown in Figure 6.

Then, all the participants reconstruct the masking data; then, the raw data can be obtained by executing the XOR operation to all the ciphertexts normally, and the faulty p_2 can return to normal.

p_1 reconstructs masking data.

$$\begin{aligned} e_1^1 &= h(k_{13} \mid 1) \oplus h(k_{14} \mid 1), \\ e_1^2 &= h(k_{13} \mid 2) \oplus h(k_{14} \mid 2), \\ e_1^3 &= h(k_{13} \mid 3) \oplus h(k_{14} \mid 3), \\ e_1^4 &= h(k_{13} \mid 4) \oplus h(k_{14} \mid 4). \end{aligned} \tag{14}$$

Ciphertext $c_1 = e_1^1 \mid e_1^2 \mid e_1^3 \oplus m_1 \mid e_1^4$.The faulty participant p_2 who applied for online restoration uses the new session keys to reconstruct the masking data:

$$\begin{aligned} e_2^1 &= h(k_{24} \mid 1) \oplus h(k_{23} \mid 1), \\ e_2^2 &= h(k_{24} \mid 2) \oplus h(k_{23} \mid 2), \\ e_2^3 &= h(k_{24} \mid 3) \oplus h(k_{23} \mid 3), \\ e_2^4 &= h(k_{24} \mid 4) \oplus h(k_{23} \mid 4), \end{aligned} \tag{15}$$

Ciphertext $c_2 = e_2^1 \oplus m_2 \mid e_2^2 \mid e_2^3 \mid e_2^4$.

p_3 reconstructs the masking data:

$$\begin{aligned} e_3^1 &= h(k_{13} \mid 1) \oplus h(k_{23} \mid 1), \\ e_3^2 &= h(k_{13} \mid 2) \oplus h(k_{23} \mid 2), \\ e_3^3 &= h(k_{13} \mid 3) \oplus h(k_{23} \mid 3), \\ e_3^4 &= h(k_{13} \mid 4) \oplus h(k_{23} \mid 4), \end{aligned} \tag{16}$$

Ciphertext $c_3 = e_3^1 \mid e_3^2 \mid e_3^3 \mid e_3^4 \oplus m_3$.

p_4 reconstructs the masking data:

$$\begin{aligned} e_4^1 &= h(k_{14} \mid 1) \oplus h(k_{24} \mid 1), \\ e_4^2 &= h(k_{14} \mid 2) \oplus h(k_{24} \mid 2), \\ e_4^3 &= h(k_{14} \mid 3) \oplus h(k_{24} \mid 3), \\ e_4^4 &= h(k_{14} \mid 4) \oplus h(k_{24} \mid 4). \end{aligned} \tag{17}$$

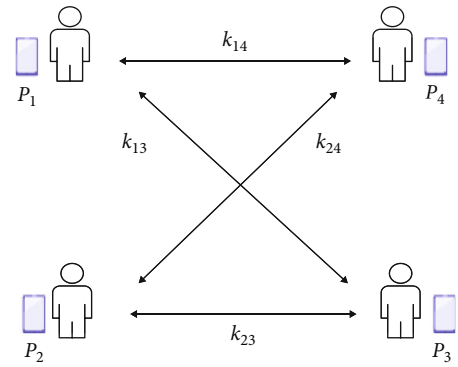

Figure 6: Session keys when the fault participants reconnect.

Ciphertext $c_4 = e_4^1 \mid e_4^2 \oplus m_4 \mid e_4^3 \mid e_4^4$.In the end, each participant publishes the ciphertext of his/her data collection phase to FN, and then, FN executes the XOR operation to all the ciphertexts, that is, $(c_1 \oplus c_2 \oplus c_3 \oplus c_4)$, to get $m_2 \mid m_4 \mid m_1 \mid m_3$.

We can use a labeled dataset which contains the electricity usage data within n days of the customers to train a model by analyzing the electricity consumption pattern of the customers for a certain period of time. The n-source anonymity data collection process can be performed n times in total to complete the collection of the n days of historical electricity consumption data of these participants. For the missing value caused by an occasional failure, we use the average value of all electricity consumption data of the participant to fill the missing value at the time of the failure.

4.4. Detector Design and Training Phase

4.4.1. Data Preprocessing. Electricity consumption data often contains missing or erroneous values. This is mainly caused by various reasons such as the unreliable transmission of measurement data and the failure of smart meters. We use the forward interpolation method to recover the missing values as

$$f(x_i) = \begin{cases} 0, & x_i \in \text{NaN}, i = 1, \\ x_{i-1,} & x_i \in \text{NaN}, i > 1, \\ x_{i,} & x_i \notin \text{NaN}. \end{cases} \tag{18}$$

where x_i stands for the value in the electricity consumption data over a period (e.g., a day). If x_i is a null or a nonnumeric character, we set it as a member of NaN (NaN is a set).

An electricity consumption dataset contains a samples, and each sample contains the electricity consumption data of the sample within n days. We randomly divide 80% of the data as the training set and 20% as the test set. In the electricity consumption dataset, there are often large differences in the electricity consumption of some users for certain days. In order to eliminate the impact of data differences on the prediction results, we use the z-score standardization to keep the values of the training samples on the same scale. In addition, the electricity consumption dataset contains the users' electricity consumption data for many days, which will cause

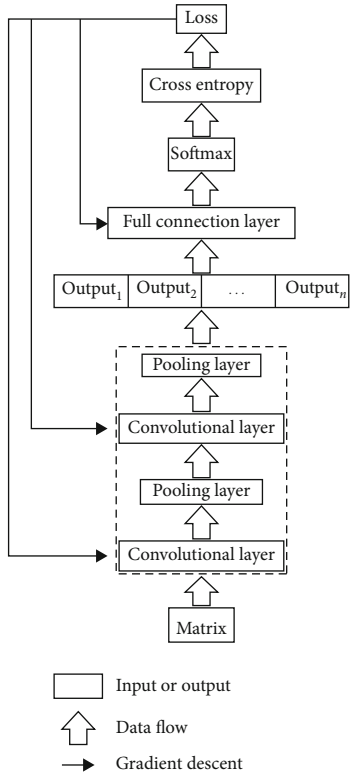

Figure 7: CNN model training flow chart.

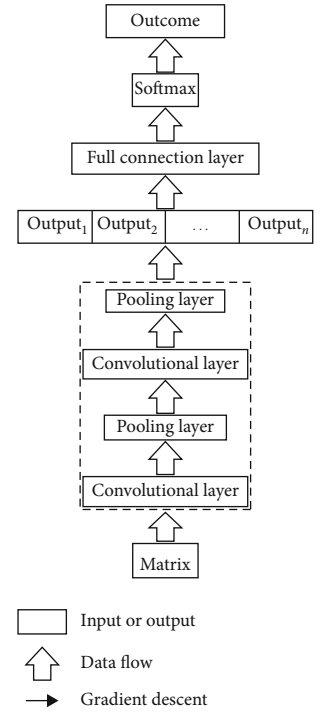

Figure 8: Model classification data flow chart.

a heavy training burden. We use PCA to reduce the dimensionality of the training set while maintaining the amount of information carried in the training set, thus ensuring that under the premise of accuracy, the training burden is reduced.

To meet the input matrix format of CNN, we can transform the 1D vector after performing PCA to a matrix C like (p, q, d), where p represents the number of rows in the matrix, q represents the number of columns in the matrix, and d represents the number of matrices. So, we can get a matrix for a single user; the shape is $(p, q, 1)$; it is shown as below:

$$\begin{pmatrix} [C_{1,1}] & \cdots & [C_{1,q}] \\ \vdots & \ddots & \vdots \\ [C_{p,1}] & \cdots & [C_{p,q}] \end{pmatrix}. \tag{19}$$

4.4.2. Our CNN Model. We use 2D convolutional layers and pooling layers and a full connection layer to build our CNN framework, as shown in Figure 7, which includes 3 stages.

(1) The shape of input data should be $(j, c, 1)$ since the target is a single user

(2) We stack the convolution layer and the pooling layer alternately to extract more features and reduce computation. We use the padding method during the convolution and pooling process. The convolution layer doubles the number of features, and the pooling layer changes the shape. For example, assume the current shape is (j, c, r); after the convolution layer, it becomes (j, c, α); after the pooling layer, it becomes $(j/2, c/2, \alpha)$

(3) We change the shape to one dimension before the full connection layer. Then, we use a full connection layer whose length is λ to change the shape to (λ). Through the softmax function, the final output shape is (2). One is the probability of theft, the other is the probability of normal, and the sum of the two probabilities is 1. If the probability of theft is greater than the normal probability, we think electricity consumption data is abnormal, and the opposite is the same. It is worth mentioning that the above variables are adjustable in order to improve the performance of our CNN model

4.4.3. Data Detection Scheme. Different from the model training process, using a trained model for data detection requires only one forward propagation, as shown in Figure 8, which includes 4 stages.

(1) Through the convolutional layer, the model extracts the preliminary features of the user's electricity consumption data

(2) Then, the model reduces the number of features and screens out the main features through the pooling layer

(3) After that, the model integrates the extracted features through the full connection layer

(4) Through the softmax function, the categories can be directly output, 0 represents the normal participant, 1 represents the suspected electricity theft participant

5. Privacy Analysis

In this section, we analyze the privacy of user data during the detection process. In the process of data detection, the detector requires raw data to ensure the accuracy of detection. From the perspective of protecting user privacy, the detector should not be able to track the source of the data.

Before publishing the data to the detector, according to the needs of the detector, we use n-source anonymity to make the data private. The n-source anonymity method guarantees the rawness and unlinkability of the data. Therefore, collecting data through the n-source anonymity method can not only enable the detector to detect the data normally but also ensure the privacy of user data.

In a word, the realization of n-source anonymity is equivalent to the realization of the privacy of the data detection process. For details about the rawness and unlinkability of the n-source anonymity, the readers can refer [21]. The data collection process in our data detection scheme is based on the n-source anonymity method, so the privacy of users is guaranteed.

6. Experiment and Analysis

In this section, we evaluate the proposed FPETD scheme by conducting experiments on a 64-bit computer with Intel (R) Core (TM) i5-6500 CPU, 3.2 GHz, 8 GB RAM, using Python, TensorFlow, and Keras framework.

6.1. Experimental Data. We use the labeled database from the State Grid Corporation of China (SGCC) [34] to conduct experiments. The SGCC dataset contains the energy usage data of 42372 customers within 1035 days, and the last column of the dataset is the label corresponding to the user, which is a single value (0 or 1): 0 represents the normal user and 1 represents the suspected electricity theft user. We randomly divide 80% as the training set and 20% as the test set. Then, we use the z-score standardization to keep the values of the training samples on the same scale. Since the samples' features have more than a thousand dimensions, which are high dimensional features, we use the PCA algorithm to reduce the samples' dimensions.

As shown in Figure 9, the abscissa of the curve represents the feature dimension. When the abscissa is 256, the ordinate of the corresponding curve has a cumulative explainable variance ratio of 99%. That is, it can still maintain 99% of the feature information when the dataset is reduced to 256 dimensions. Then, we reshape the 256-dimensional features into a matrix whose shape is (16, 16, 1). The data is transformed into the matrix as below:

$$\begin{pmatrix} [C_{1,1}] & \cdots & [C_{1,16}] \\ \vdots & \ddots & \vdots \\ [C_{16,1}] & \cdots & [C_{16,16}] \end{pmatrix}. \tag{20}$$

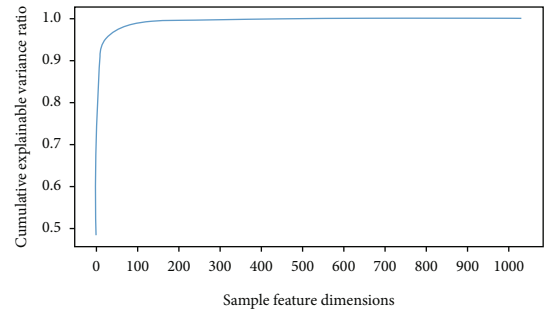

Figure 9: Cumulative explainable variance.

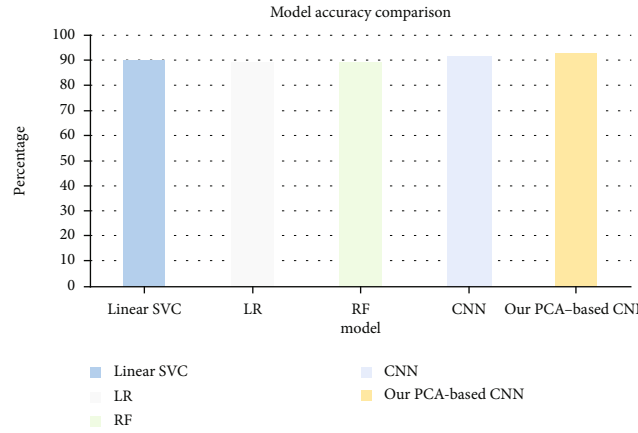

Figure 10: Model accuracy comparison.

6.2. Model Training Phase. Our convolutional neural network contains 2 convolutional layers, 2 pooling layers, and 1 full connection layer. Our first convolutional layer uses 32 convolution kernels with a size of (5, 5), and the sliding step size is (1, 1), using the method of padding when doing convolution operation in the input matrix. The output of the first convolutional layer is a matrix whose shape is (16, 16, 32), and then, the matrix is passed to the first pooling layer. We adopt the maximum pooling method, the sliding window size of the pooling layer is set to the size of (2, 2), and the sliding step size is set to (2, 2).

The output of the first pooling layer is a matrix whose shape is (8, 8, 32), and then, the matrix is passed to the second convolution layer which has 64 convolution kernels with a size of (5, 5), and the sliding step size is (1, 1). Using the method of padding when doing convolution operation in the input matrix, the output of the second convolution layer is a matrix whose shape is (8, 8, 64), and then, the matrix is passed to the second pooling layer. Using the maximum pooling method, the sliding window size of the pooling layer is set to the size of (2, 2), the sliding step size is set to (2, 2), and the output of the second pooling layer is a matrix whose shape is (4, 4, 64). Then, expand the matrix into a (1, 1024) vector. After that, it is passed to the full connection layer to synthesize the previous features, and the category probabilities are outputted through the softmax function.

The final predicted categories are compared with the true categories. Through backpropagation, the parameters in the

Table 1: Property comparison.

	Proposed scheme	Yao et al. [6]	Zheng et al. [7]	Jindal et al. [8]
Technique adopted	CNN	CNN	CNN	Decision tree and SVM
User privacy	Yes	No	No	No
Dispatching of smart grid	Yes	Yes	No	Yes
Massive data processing	Yes	Yes	Yes	No

convolutional neural network are iterated in backpropagation. They are updated again and again, infinitely approaching our real parameters. This is the model training process.

6.3. Model Evaluation. We use the accuracy score as a performance score to evaluate the performance of the trained PCA-based CNN model. The equation of accuracy score performs as below:

$$\text{accuracy}(\hat{y}, y) = \frac{1}{n}\sum_{i=1}^{n}\delta(y_i, \hat{y}_i), \tag{21}$$

where

$$\delta(y_i, \hat{y}_i) = \begin{cases} 1, & y_i = \hat{y}_i, \\ 0, & \text{else}. \end{cases} \tag{22}$$

The $\hat{y}$ represents the predicted value, and the y represents the true value. $\hat{y}_i$ is the predicted value of the i-th sample and y_i is the corresponding true value; n represents the number of samples.

We use the z-score standardization and PCA to preprocess the test set. We train the model for 100 epochs to update the model parameters. After training the model, we evaluate the model on the test set; the model accuracy score is 0.9286.

6.4. Model Comparison. We compared the designed PCA-based CNN model to the single CNN model and other traditional machine learning methods, such as linear SVC [35], logistic regression (LR) [36], and random forest (RF) [37]. The experiment shows that the accuracy score of our PCA-based CNN deep learning model is better than that of the single CNN model and other traditional machine learning models, which proves that our dimensionality reduction method greatly improves the accuracy of electricity theft detection, as shown in Figure 10.

6.5. Comparison with Existing Schemes. This section mainly describes the comparison with existing electricity theft detection schemes. In the scheme of Yao et al. [6], it is assumed that the SG detector is trusted, but in reality, the detector is often untrusted, so it may leak user privacy. In Zheng et al. [7] and Jindal et al. [8], there is also a problem of leaking user privacy. Our FPETD scheme is to treat the detector as completely untrusted; after collecting the data, CS directly adds these raw data; therefore, CS can know the overall power demand of the building and then make decisions about the power distribution of the building.

As shown in Table 1, our FPETD scheme can protect user privacy and detect big data and also obtain the sum power consumption data to make power distribution decisions.

7. Conclusion

In this paper, we propose the FPETD scheme to realize real-time electricity theft detection in the smart grid. In our scheme, we designed a fault-tolerant raw data collection protocol to collect electricity data and cut off the correspondence between users and their data, thereby ensuring the fault tolerance and data privacy during the electricity theft detection process. Experiments have proven that our dimensionality reduction method before training the model makes the accuracy of our model better than others. However, the computational burden of our data collection process is a bit heavy. In our future work, we consider reducing the computational burden in the data collection process.

Data Availability

The labeled data from the State Grid Corporation of China (SGCC) were used to support this study.

Conflicts of Interest

The authors declare that there is no conflict of interest.

Acknowledgments

The study of the manuscript is funded by the National Natural Science Foundation of China (62072133, 61662016) and Key Projects of the Guangxi Natural Science Foundation (2018GXNSFDA281040).

References

[1] N. O. Shokoya and A. K. Raji, "Electricity theft: a reason to deploy smart grid in South Africa," in *2019 International Conference on the Domestic Use of Energy (DUE)*, pp. 96–101, Wellington, South Africa, 2019.

[2] B. C. Neagu and G. Grigoras, "Decision-making approach for choosing of electricity supplier to improve the energy efficiency," in *2019 International Conference on ENERGY and ENVIRONMENT (CIEM)*, pp. 343–347, Timisoara, Romania, 2019.

[3] J. Presnal, H. Houston, and G. Maberry, "The electrical safety program and the value in partnering with health & safety professionals," in *2020 IEEE IAS Electrical Safety Workshop (ESW)*, pp. 1–7, Reno, NV, USA, 2020.

[4] B. S. England and A. T. Alouani, "Multiple loads-single smart meter for measurement and control of smart grid," in *2019 IEEE Innovative Smart Grid Technologies - Asia (ISGT Asia)*, pp. 2440–2444, Chengdu, China, 2019.

[5] Z. Gaofeng, H. Xuemin, L. Pengxi et al., "Application and research of enterprise-level business and data fusion data analysis service platform based on big data technology," in *2019 IEEE 5th International Conference on Computer and Communications (ICCC)*, pp. 1950–1954, Chengdu, China, 2019.

[6] D. Yao, M. Wen, X. Liang, Z. Fu, K. Zhang, and B. Yang, "Energy theft detection with energy privacy preservation in the smart grid," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 7659–7669, 2019.

[7] Z. Zheng, Y. Yang, X. Niu, H. Dai, and Y. Zhou, "Wide and deep convolutional neural networks for electricity-theft detection to secure smart grids," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 4, pp. 1606–1615, 2018.

[8] A. Jindal, A. Dua, K. Kaur, M. Singh, N. Kumar, and S. Mishra, "Decision tree and SVM-based data analytics for theft detection in smart grid," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 3, pp. 1005–1016, 2016.

[9] R. Punmiya and S. Choe, "Energy theft detection using gradient boosting theft detector with feature engineering-based preprocessing," *IEEE Transactions on Smart Grid*, vol. 10, no. 2, pp. 2326–2329, 2019.

[10] N. Takbiri, A. Houmansadr, D. L. Goeckel, and H. Pishro-Nik, "Privacy of dependent users against statistical matching," *IEEE Transactions on Information Theory*, vol. 66, no. 9, pp. 5842–5865, 2020.

[11] K. Renuka, S. Kumar, S. Kumari, and C.-M. Chen, "Cryptanalysis and improvement of a privacy-preserving three-factor authentication protocol for wireless sensor networks," *Sensors*, vol. 19, no. 21, p. 4625, 2019.

[12] C. D. Shin, K. K. Joo, J. H. Seo, and Z. Atif, "Study on reactor neutrino directionality search utilizing vertex information reconstructed by PMT operating state in a liquid scintillator detector," *IEEE Transactions on Nuclear Science*, vol. 67, no. 9, pp. 1996–2002, 2020.

[13] Y. Liu, W. Guo, C. Fan, L. Chang, and C. Cheng, "A practical privacy-preserving data aggregation (3PDA) scheme for smart grid," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 3, pp. 1767–1774, 2019.

[14] J. Song, Y. Liu, J. Shao, and C. Tang, "A dynamic membership data aggregation (DMDA) protocol for smart grid," *IEEE Systems Journal*, vol. 14, no. 1, pp. 900–908, 2020.

[15] M. S. Ballal, H. Suryawanshi, M. K. Mishra, and G. Jaiswal, "Online electricity theft detection and prevention scheme for smart cities," *IET Smart Cities*, vol. 2, no. 3, pp. 155–164, 2020.

[16] S. Mujeeb, N. Javaid, R. Khalid, M. Imran, and N. Naseer, "DE-RUSBoost: an efficient electricity theft detection scheme with additive communication layer," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pp. 1–6, Dublin, Ireland, 2020.

[17] A. Asrari, M. Ansari, J. Khazaei, and P. Fajri, "A market framework for decentralized congestion management in smart distribution grids considering collaboration among electric vehicle aggregators," *IEEE Transactions on Smart Grid*, vol. 11, no. 2, pp. 1147–1158, 2020.

[18] M. Murakami, M. Matsuno, S. Okamoto, and N. Yamanaka, "Experimental evaluation of application triggered flow classification using operated data center traffic data," in *2019 24th OptoElectronics and Communications Conference (OECC) and 2019 International Conference on Photonics in Switching and Computing (PSC)*, pp. 1–3, Fukuoka, Japan, 2019.

[19] F. Haider, S. Pollak, P. Albert, and S. Luz, "Extracting audio-visual features for emotion recognition through active feature selection," in *2019 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 1–5, Ottawa, ON, Canada, 2019.

[20] Y. Zhang, Q. Chen, and S. Zhong, "Privacy-preserving data aggregation in mobile phone sensing," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 5, pp. 980–992, 2016.

[21] Y. Liu, Y. Wang, X. Wang, Z. Xia, and J. Xu, "Privacy-preserving raw data collection without a trusted authority for IoT," *Computer Networks*, vol. 148, pp. 340–348, 2019.

[22] J. Chen, G. Liu, and Y. Liu, "Lightweight privacy-preserving raw data publishing scheme," *IEEE Transactions on Emerging Topics in Computing*, p. 1, 2020.

[23] A. Rehman, A. Khan, M. A. Ali, M. U. Khan, S. U. Khan, and L. Ali, "Performance analysis of PCA, sparse PCA, kernel PCA and incremental PCA algorithms for heart failure prediction," in *2020 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*, pp. 1–5, Istanbul, Turkey, 2020.

[24] G. Chen, Y. Chen, Z. Yuan, X. Lu, X. Zhu, and W. Li, "Breast cancer image classification based on CNN and bit-plane slicing," in *2019 International Conference on Medical Imaging Physics and Engineering (ICMIPE)*, pp. 1–4, Shenzhen, China, 2019.

[25] S. Chen, W. Sun, L. Huang, X. Yang, and J. Huang, "Compressing fully connected layers using Kronecker tensor decomposition," in *2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT)*, pp. 308–312, Dalian, China, 2019.

[26] X. Li, Z. Hu, and X. Huang, "Combine ReLU with Tanh," in *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, pp. 51–55, Chongqing, China, 2020.

[27] M. Wang, S. Lu, D. Zhu, J. Lin, and Z. Wang, "A high-speed and low-complexity architecture for softmax function in deep learning," in *2018 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, pp. 223–226, Chengdu, China, 2018.

[28] I. Arora and A. Saha, "Comparison of back propagation training algorithms for software defect prediction," in *2016 2nd International Conference on Contemporary Computing and Informatics (IC3I)*, pp. 51–58, Greater Noida, India, 2016.

[29] C. M. Chen, Y. Huang, K. H. Wang, S. Kumari, and M. E. Wu, "A secure authenticated and key exchange scheme for fog computing," *Enterprise Information Systems*, vol. 1, pp. 1–16, 2020.

[30] Y. Lin and H. Shen, "Cloud fog: towards high quality of experience in cloud gaming," in *2015 44th International Conference on Parallel Processing*, pp. 500–509, Beijing, China, 2015.

[31] L. Lyu, K. Nandakumar, B. Rubinstein, J. Jin, J. Bedo, and M. Palaniswami, "PPFA: privacy preserving fog-enabled aggregation in smart grid," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3733–3744, 2018.

[32] A. Sanjab, W. Saad, I. Guvenc, A. Sarwat, and S. Biswas, "Smart grid security: threats, challenges, and solutions," 2016, https://arxiv.org/abs/1606.06992.

[33] C.-M. Chen, B. Xiang, T.-Y. Wu, and K.-H. Wang, "An anonymous mutual authenticated key agreement scheme for wearable sensors in wireless body area networks," *Applied Sciences*, vol. 8, no. 7, p. 1074, 2018.

[34] https://www.datafountain.cn.

[35] H. Li and B. Liu, "Loss analysis simulation of SVC/DC deicer under SVC mode," in *2016 IEEE PES Asia-Pacific Power and Energy Engineering Conference (APPEEC)*, pp. 2564–2568, Xi'an, China, 2016.

[36] D. W. Hosmer, T. Hosmer, S. Le Cessie, and S. Lemeshow, "A comparison of goodness-of-fit tests for the logistic regression model," *Statistics in Medicine*, vol. 16, no. 9, pp. 965–980, 1997.

[37] V. Svetnik, A. Liaw, C. Tong, J. C. Culberson, R. P. Sheridan, and B. P. Feuston, "Random forest: a classification and regression tool for compound classification and QSAR modeling," *Journal of Chemical Information and Computer Sciences*, vol. 43, no. 6, pp. 1947–1958, 2003.