A Stacked Machine and Deep Learning-Based Approach for Analysing Electricity Theft in Smart Grids

Inam Ullah Khan, Nadeem Javeid, C. James Taylor, Kelum A. A. Gamage, and Xiandong Ma

***Abstract*—The role of electricity theft detection (ETD) is critical to maintain cost-efficiency in smart grids. However, existing methods for theft detection can struggle to handle large electricity consumption datasets because of missing values, data variance and nonlinear data relationship problems, and there is a lack of integrated infrastructure for coordinating electricity load data analysis procedures. To help address these problems, a simple yet effective ETD model is developed. Three modules are combined into the proposed model. The first module deploys a combination of data imputation, outlier handling, normalization and class balancing algorithms, to enhance the time series characteristics and generate better quality data for improved training and learning by the classifiers. Three different machine learning (ML) methods, which are uncorrelated and skillful on the problem in different ways, are employed as the base learning model. Finally, a recently developed deep learning approach, namely a temporal convolutional network (TCN), is used to ensemble the outputs of the ML algorithms for improved classification accuracy. Experimental results confirm that the proposed framework yields a highly-accurate, robust classification performance, in comparison to other well-established machine and deep learning models and thus can be a practical tool for electricity theft detection in industrial applications.**

***Index Terms*—Electricity theft detection, big data, preprocessing, data classification, smart grid.**

I. Introduction

ONE OF the main goals of a smart grid is to decrease power system losses and to balance the gap between power supply and demand [1]. The losses which a power network encounters from generation to distribution are of two types: technical losses (TL) and non-technical losses (NTL). TL are caused during the transfer of energy in transformers, transmission lines and cables and thus cannot be simply averted in a distributed network. The NTL, in contrast, happens when electricity is used fraudulently to reduce utility charges. Such cases include meter bypassing and tampering, synchronously switching power circuits and tapping on secondary voltages [2]. The main reason for NTL in a power network is electricity theft, which presents an estimated revenue loss of $89.3 billion annually worldwide [3].

Generally, an electricity theft detection (ETD) mechanism of some form is expected because of economic and industrial requirements [4]. Also, customers have a predefined power purchase threshold, and due to NTL, the burden on end-users is ultimately increased. In recent years, the reduction of NTL has become one of the leading drivers in smart grid and the use of advanced methods, such as big data analysis, is becoming standard for detecting anomalous power consumption. By controlling electricity theft, utilities curtail expenditure on energy and are able to better control the power demand for a specific period. This yields financial benefits in terms of generation cost and helps to control a broad range of irregularities both at the planning and distribution levels [5]. A precise and efficient theft detection method reduces the supply-demand gap and helps ensure a stable and efficient power management system. It addresses uncertain power generation challenges and brings higher reliability to the available energy sources. However, the ETD phenomenon is dynamic and complex in nature, comprising diverse aspects of energy consumption and the variation tendencies over time are nonlinear. Electricity demand is influenced by numerous features, such as inherent demand, fuel price, renewable energy supply and transmission, with hourly variations. Since electricity demand alters recurrently and a large number of smart meters monitor the associated factors, all in real-time, the data volume produced by smart sensors and smart sub-meters is enormous and difficult to analyse [6]–[8].

There are three major challenges in supervised NTL detection methods, i.e., handling missing and outliers' data values during data preprocessing, data class unbalancing and choosing an appropriate classifier.

In the first instance, feature pre-processing is fundamental to the application of the classifier. In a study [9], the authors utilized the conjunction of support vector machine (SVM) and decision tree (DT) algorithms to detect electricity theft with higher accuracy. The study yields very promising results; however, the issue of recovering missing values from existing data has remained unattended. Without addressing the missing data issue, the scalability and robustness of the proposed model

Manuscript received May 20, 2021; revised October 12, 2021; accepted December 3, 2021. Date of publication December 9, 2021; date of current version February 21, 2022. This work was supported in part by Lancaster University, U.K.; in part by COMSATS University Islamabad (Lahore Campus); and in part by the U.K. Engineering and Physical Sciences Research Council (EPSRC) under Grant EP/R02572X/1. Paper no. TSG-00786-2021. *(Corresponding author: Xiandong Ma.)*

Inam Ullah Khan, C. James Taylor, and Xiandong Ma are with the Engineering Department, Lancaster University, Lancaster LA1 4YW, U.K. (e-mail: xiandong.ma@lancaster.ac.uk).

Nadeem Javeid is with the Department of Computer Science, COMSATS University Islamabad, Islamabad 44000, Pakistan.

Kelum A. A. Gamage is with the James Watt School of Engineering, University of Glasgow, Glasgow G12 8QQ, U.K.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TSG.2021.3134018.

Digital Object Identifier 10.1109/TSG.2021.3134018

cannot be guaranteed. The authors of [10] conducted a detailed review of 34 supervised machine learning (ML) based research articles on ETD and found that only half of the considered research articles addressed the issue of missing data values. Aslam *et al.* [11] used SVM and XGBoost as a boosting classifier for anomalies detection in improving the operational safety of a power network. Based on load profiles and available auxiliary data information, the data were analysed to learn key features and characteristics for consumers ranking in accordance with their electricity usage patterns. First, the training process of the SVM classifier started by minimizing empirical risk minimization. Afterwards, the boosting algorithm is employed to associate correct categories to improve the final prediction accuracy. However, the data preparation steps were not considered and the presence of various outliers in primary data from the market can make the classification accuracy volatile. Our paper has detailed addressed the mentioned problem in subsequent sections.

Data class unbalancing is another critical problem in smart meters' labelled datasets for ETD applications. It causes a biasness problem because the prediction model might learn key features and concepts related to the majority class and minority class samples (theft cases) would most often remain unattended. To obtain an efficient and unbiased ML model performance, an equal representation of both data samples is essentially required. Li *et al.* [12] used the convolutional neural network (CNN) for accurate identification of electricity fraud. However, an important issue of model generalization can occur in CNN when the final prediction result is obtained from the fully connected layer. This issue is resolved by [13] when the authors employ a random forest (RF) algorithm for obtaining the classification task final output. In their work, a well-known class balancing method, synthetic minority oversampling technique (SMOTE) is used to increase prediction model's capabilities to learn key features and characteristics of theft cases. However, SMOTE generates synthetic samples for minority class that may lead creation of an overfitting problem. This means that the proposed model is performing better on seen/training data but its performance degrades for unseen/test data [1].

Once the process of data preprocessing is completed, the next challenge is the selection of an appropriate classifier to efficiently segregate the honest and theft consumers. In more general terms, hardware and non-hardware solutions are two main ways for electricity theft prediction. Non-hardware solutions are classification algorithms, for which ML and deep learning (DL) methods, such as SVM, DT, RF, artificial neural networks (ANN) and generative adversarial networks (GAN) are very popular [14]–[18]. However, a large body of literature suggests that none of these approaches is perfect and each method may exhibit its drawbacks during the classification procedure. Big data characteristics such as high volume, high velocity and high veracity are creating new challenges and require new processing paradigms. For example, SVM is computationally expensive because of the large number of support vectors for large training datasets, and the associated hyper-parameters can be problematic to tune. Similarly, DT and RF usually face over-fitting problems. The ANN and GAN convergences cannot be easily controlled, and these methods have limited generalization capabilities. In the context of electricity theft prediction, the challenging is to improve robustness, scalability and accuracy in the face of widespread nonlinear data.

In the present work, we investigate various ETD issues, including binary classification tasks where the main objective is to predict the normal and fraudulent patterns of customers. ML methods provide the underpinning framework. During the classification process, each ML method attempts to separate different data points and explain a class value. Although SVM, RF and DT are promising approaches, they may outperform each other or have defects in different cases. Thus, the following challenges must be addressed when making an accurate prediction between the two patterns.

- *Highly imbalanced theft data*: One of the main problems in the real-world dataset is imbalanced classes [14]. This is the scenario where non-fraudulent samples far outweigh the fraudulent ones. The common methods to deal with the imbalanced class distribution problem is random oversampling and under-sampling. However, both methods have known drawbacks that cause the supervised ML models to become bias and overfit towards majority class samples, thus leading to inaccurate prediction results for theft cases.
- *Difficulty in parameters tuning*: In ML methods, numerous hyperparameters control the learning process. There is no analytical formula available to calculate an appropriate value of these hyperparameters, which affect the performance of models in the classification task. Gradient descent and cross-validation [19] are two common methods to adjust hyperparameters. However, both methods increase the computational complexity and make the converging process difficult.
- *High computational overhead*: According to [1], [19], deep learning (DL) methods are weak to process uncertain information and have high computational costs. In electricity theft prediction process, the presence of redundant and extraneous features increases computational complexity and makes the final classifier's training process hard and prevents it from being a good fit model, which decreases the prediction accuracy.

To address the above, we propose a new integrated data preparations, first and second-order classification (PFSC) framework, as summarised in Fig. 1. The three components of PFSC are: data preparation based on interpolation, outliers detection, normalization, and balancing (IONB) tasks; a first-order ML classifier based on SVM, RF and gradient boosting decision tree (GBDT) methods; and a second-order classification step using a temporal convolutional network (TCN). Specifically, the missing values in the data are filled by applying an interpolation method to achieve data uniformity. Subsequently, outlier handling and data normalization steps are used to set the values between 0-1 and to ensure data consistency. In the State Grid Corporation of China (SGCC) dataset, more honest (91%) and less dishonest (9%) consumers exist. Thus, the final task of IONB is to apply the sampling technique to get an equal distribution of both classes. Once the data preparation

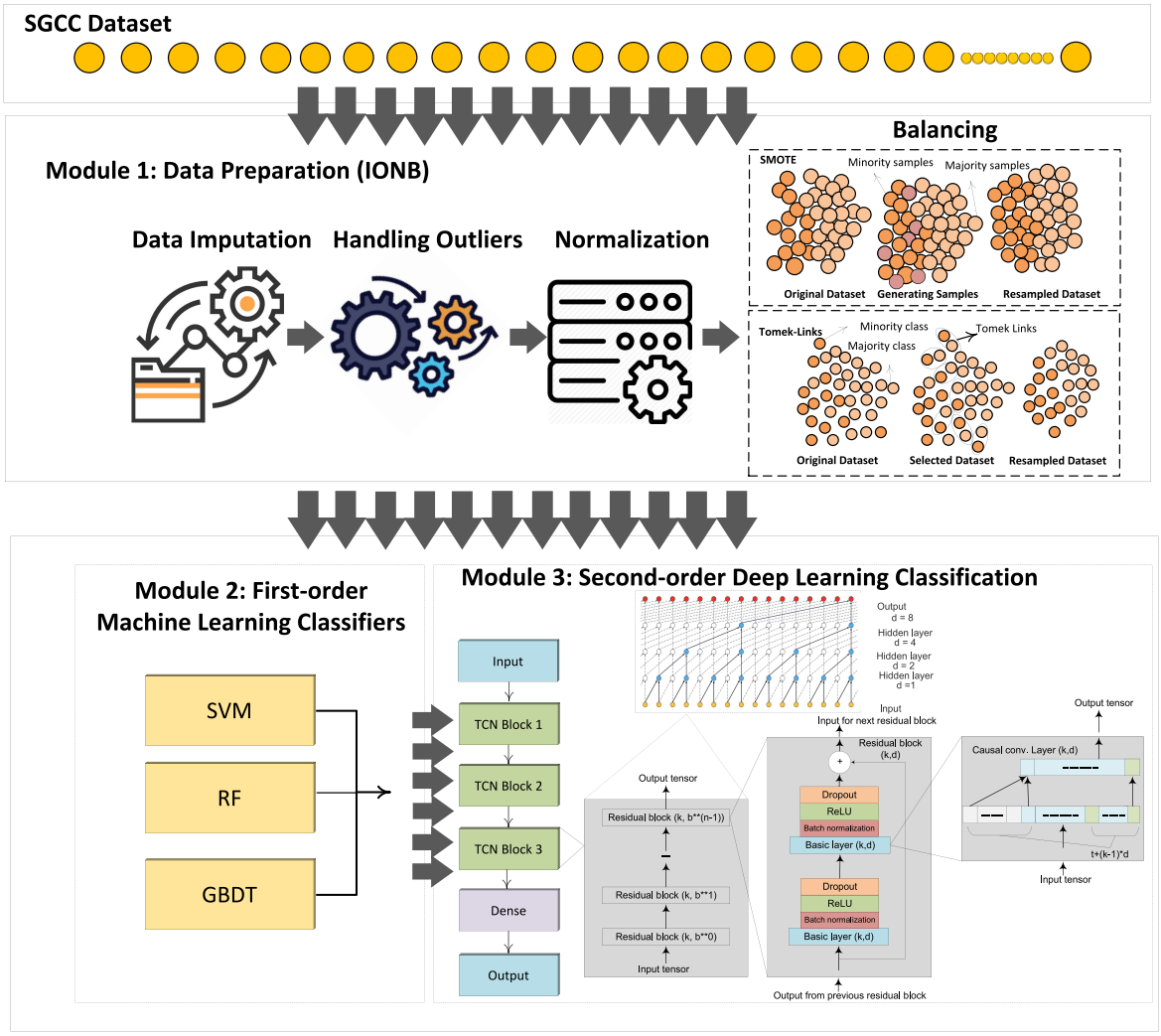


Fig. 1. Proposed framework for electricity theft detection.

task is performed, the prepared data are used to train three different classifiers to construct a first-order ML classification model. It is natural to expect that multiple methods will lead to superior performance [19]–[22]. Hence, the outputs of the three ML classifiers are stacked and provided to the DL method (a recently developed second-order classifier in our case) to obtain the final classification. Our recent conference article [23] also proposes an integrated data pre-processing and resampling methods and present some preliminary results. The present work builds on this concept but uses a new approach to classifiers. In this manner, the main contributions of the research are:

- Development of an integrated ETD framework to achieve accurate theft detection using real data in a smart grid. To our knowledge, this represents a first attempt to integrate data preparation steps with first and second-order classifiers into a single framework for the studied problem. Due to cascading effects, real smart meter data are efficiently handled and analyzed.
- An extensive IONB approach is proposed, involving imputation, handling outliers, normalization and class balancing algorithms for better training of classifiers. The original dataset has a sample size of 42372 and each sample has 1035 features, with issues like redundancy and irrelevancy. These issues can be problematic for both the ML and DL models. As suggested in the literature [1], [5], [6], [19], ML models have lower computational overheads when trained in the presence of such big data. In our paper, the main aim is to achieve higher prediction accuracy. However, there is always a trade off between accuracy and computational complexity. Both higher accuracy and computational efficiency are difficult to attain simultaneously. The multi-model ensemble method trains a second-order DL classifier on the limited predicted features provided by the first-order ML classifiers. It is important to note that the first-order classifier training process is conducted in parallel and there is a negligible execution time difference between them. The second-stage classifier (ensembler) optimally combines the first-order models' predictions (only three features) to provide final results, with higher accuracy and minimum computational complexity.
- Extensive simulations based on real-world data traces from electric grid's workload have been investigated for

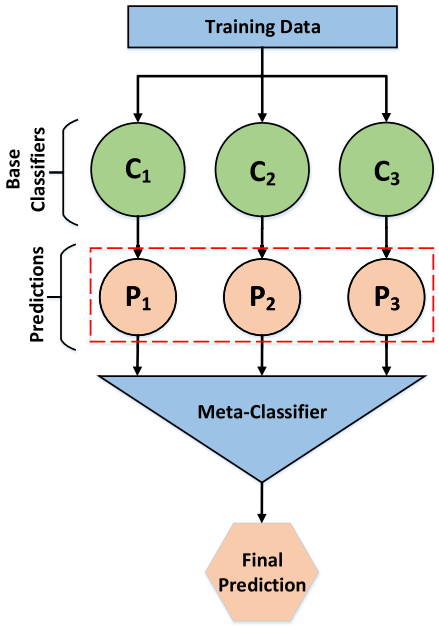


Fig. 2. Schematic diagram of the proposed system.

performance assessment. The experimental results confirm that the DL based multi-model ensemble method makes efficient use of multi-variate time sequence data and offers high accurate predictions than any single machine and deep learning model trained in isolation.

The remainder of this paper is organised as follows. Section II describes the proposed theft detection framework. Section III presents the data preparation module. Section IV develops the base and meta-classifier procedures. The experimental results for several realistic case studies are explained in Section V. Finally, conclusions are presented in Section VI.

II. System Framework Overview

The basic problem in ETD is to improve accuracy. Various factors can impact the electricity consumption pattern of the consumers, which makes classifier training challenging. To improve the accuracy of the proposed PFSC framework, we develop a sequential IONB, a first-order ML classifier and a DL-based second-order classifier for the final prediction of normal and fraudulent patterns.

The approach begins with raw data standardisation, the first module in Fig. 1. Standardization is pivotal for the implementation of the whole framework. In the second module, the standardized data are fed into the base classifiers to train SVM, RF and GBDT in parallel. The schematic diagram shown in Fig. 2 illustrates how base classifiers perform predictions. Due to the decoupling design of the selection algorithm, the process could execute in a distributive fashion. Finally, in the third module of Fig. 1, the processed data are sent to build the DL model, namely the TCN. We prefer TCN because of advantages to learn essential laws and key features from a large dataset. Also, it depicts stronger complex and nonlinear function fitting and computing abilities than shallow ML models, hence make it more suitable choice for classification tasks [24]. The details of these modules are described in next two sections.

III. IONB-Based Sequential Data Preparations

Data preparation is often the first important step while analyzing big data problems. It ensures accuracy in the data which leads to accurate insights and better classifier training. We propose a sequential IONB method on collected data to ensure accurate quantifications, i.e., true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN) found in a confusion matrix (CM). The sequential procedure starts with imputation, handling outliers, data normalization and finally handling the class imbalanced problem. We assume a matrix,

$$\mathrm{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix} = \begin{bmatrix} \overrightarrow{t_1} \\ \overrightarrow{t_2} \\ . \\ . \\ . \\ \overrightarrow{t_m} \end{bmatrix}, \tag{1}$$

where,

$$\overrightarrow{t_k} = [x_{k1}, x_{k2}, \ldots, x_{kn}] k \in [1, m]. \tag{2}$$

to represent electricity consumption pattern. Time stamps and the feature index of recorded data are represented by the rows and columns, respectively. The index, i.e., x_{mn} is the $n-th$ component of the $m-th$ electricity usage values that need to be classified.

A. Recovering Missing Data

The consumption record of electricity comprise numerous missing values and incomplete information. The main reasons behind the problem may be due to data corruption and failure of hardware. In time-series data analysis, the missing values cannot be simply neglected because these values can significantly impact the performance and quality of the final predictions. A better way is to impute the missing value by calculating the mean/median of the neighboring non-missing values. In the present work, missing values are retrieved through the interpolation method in [3], as follows,

$$f(x_i) = \begin{cases} \left(\frac{x_{i-1}+x_{i+1}}{2}\right), & if\ x_i \in NaN, x_{i\pm1} \notin NaN, \\ x_i, & otherwise, \end{cases} \tag{3}$$

where x_i are the missed (null) or recorded values contain by the dataset. The null value is a non-numeric character, expressed as NaN. If the value of x_i is null, then Eq. (3) is utilized to fill the corresponding value.

Motivated by [25], we create an equal number of consumption records for each user by removing from the original dataset any clients with >600 null values. If a user contains m <7 missing samples, the linear interpolation method is used on existing data to fill missing values. Similarly, the missing values are replaced with zero if m≥7.

B. Handling Outliers

We have found numerous erroneous values in the SGCC dataset alluded to above. The presence of outliers misleads the training process, takes longer training times, resulting less accurate models and, ultimately, mediocre results. The "three-sigma rule of thumb" used in [3] is employed for mitigating the outliers and restoring the data, as shown below,

$$f(x_i) = \begin{cases} X, & if\ x_i > X, \\ x_i, & otherwise, \end{cases} \tag{4}$$

where X indicates $Avg(x_i) + 2\sigma(x_i))$ and σ represents the standard deviation of x.

C. Data Standardization

ML and DL methods are sensitive to diverse data. Hence, we perform data normalization using the Min-Max method calculated in following equation,

$$f(x_i) = \frac{X_i - min(X)}{max(X) - min(X)}. \tag{5}$$

D. Handling Imbalanced Class

One of the main problems found in the electricity theft dataset is the majority class (honest consumers) domination over the minority class (dishonest). The imbalanced data have a non-uniform distribution of target variables and this causes the classifier to become skewed towards the majority class [26], [27]. As a result, the classifier becomes bias and exhibits misleading performance towards the minority class samples (theft cases). In the ETD problem, this problem is more critical to handle because minority class samples identification is more important than the majority class (honest customers).

Hence, in this work, we develop a new class balancing method that strategically couple the characteristics of over-sampling and under-sampling methods to minimise the misclassification cost. We name the proposed technique STLU (SMOTE + Tomek link undersampling) and it is applied for the first time in this framework to adjust for the unbalanced class distribution problem.

In STLU, SMOTE is an oversampling technique, which generates new instances in the minority class synthetically by interpolating between numerous minority class samples that lie together. The creation of a synthetic data point starts by choosing a random sample from s samples. In feature space, the Euclidean distance between the random sample and its k nearest neighbors is calculated. The new synthetic sample is created when one of those neighbors' point k vector is multiplied with a random number a. The value of a lies between 0 and 1. This procedure is repeated until the distribution between both classes is balanced.

Although oversampling methods can help achieve balance class distributions, some other problems present in the electricity theft datasets, such as skewed class distributions, are not solved. More generally, some majority class samples might be invading into minority class portions due to the undefined class clusters. The opposite can also happen, i.e., when interpolation causes expansion of the minority class cluster and introduces artificial minority class samples that are too deep in the majority class area. To create well-established class clusters, Tomek links [27] between examples are recognised and these examples are then removed from the dataset.

Unlike SMOTE, the TLU method removes unwanted majority class samples from class boundaries to make an equal proportion. The TLU defines a pair of data points (x_i, x_j) in the majority class where x_i belongs to the minority class and x_j denotes the majority class sample. The distance between both samples is denoted as $d(x_i, x_j)$. The pair (x_i, x_j) forms a Tomek link when no sample x_k satisfies the condition such that $d(x_i, x_k) < d(x_i, x_j)$ and $d(x_j, x_k) < d(x_i, x_j)$. In this way, the data samples in the majority class having the least Euclidean distance with minority class samples are removed.

To combine oversampling and undersampling methods, we use an imbalanced-learn Python library [26]. The library provides a wide range of resampling methods, as well as a pipeline class to allow transformation to be stacked in sequence on a dataset. The STLU method with the help of pipeline first applies SMOTE and then TLU to the output of the oversampling transform before returning the final outcome.

Good results may be obtained when both the oversampling and undersampling methods are combined. For illustration, a moderate quantity of oversampling increases the bias towards the minority class, whilst undersampling by a modest amount can result in a decreased bias towards the majority class samples. This adoption of a combined strategy helps improve overall performance in contrast to applying one or the other method in isolation. In this work, the SGCC dataset initially consisted of 1 (minority):10 (majority) class data distribution. We first use SMOTE, which increases the ratio to 3:10 by synthetically generating minority class samples. Subsequently, TLU is used to further adjust the ratio to 1:1 by removing samples from the majority class.

The efficiency of first and second-order classifiers induced from standalone SMOTE, TLU and STLU (SMOTE+Tomek) as a pre-processing method for a highly imbalanced electricity theft dataset is evaluated in Section V-B.

IV. Classifier Adjustment

Following the IONB steps, the data are clean, formatted and transformed to train the classifier. In terms of the classifiers, we choose stacked generalization, arguably the best approach among various state-of-the-art methods, recently winning many Netflix and Kaggle competitions for classification tasks [28], [29]. This is an efficient and robust method of learning high-level classifiers (second-order) on top of the base classifiers (first-order) to achieve greater predictive accuracy. Specifically, first-order models involve three different ML methods that are established on the classification problem in a different way. A newly developed DL method is then employed as a high-level model to ensemble the output of the first-order models and achieve reliability in classification tasks.

A. Base Classifiers

In SVM [9], training data are initially mapped into a feature space of high dimensionality. With the help of a hyperplane,

TABLE I
HYPERPARAMETERS OF THE ML MODELS

Classifier	Hyperparameters	Range of values	Optimal values
SVM	Cost penalty (C), Intensive loss function (σ), kernel function (k).	C = 0.01, 0.11, 1, 10, 100. σ = 0.0001, 0.001, 0.01, 0.1, 1. k = linear, ploly, rbf, sigmoid.	C = 1, σ = 0.1, k = rbf.
RF	DT, Sample leaves (SL), Sample splits (SS), Criterion.	DT = 10, 15, 20, 25, 30. SL = 1, 5, 10, 15, 20. SS = 3, 4, 5, 6, 7. Criterion = gini, entropy	DT = 15, SL = 5 SS = 7, Criterion = gini.
GBDT	Number of estimators (NE), Maximum depth (MD), Learning rate (LR).	NE = 60, 90, 120, 150, 180. MD = 1, 3, 6, 9, 12. LR = 0.0001, 0.001, 0.01, 1, 10.	NE = 180, MD = 9, LR = 0.001.

the two categories of data are separated in such a way that the gap between different data points is largest. Tested samples are mapped implicitly to the same space and classified based on which side of the class they belong to with greater certainty.

RF [10] is an ensemble ML algorithm and has recently gained much attention on classification tasks due to out-of-the-box learning algorithms and its relative simplicity, diversity and computational capabilities. RF involves constructing a large number of decorrelated decision trees, each of which corresponds to a random vector value, sampled independently but with a similar distribution. By adopting the wisdom of the crowd, the output class is the one that receives majority votes in the forest. In contrast to RF, GBDT [11] is an ensemble technique that combines multiple DT models for building a stronger prediction model. In GBDT, DT are added one at a time in a gradual, additive and sequential fashion to reduce the prediction error of prior DT models. The models are trained using an arbitrary differential loss function and gradient descent optimization algorithm.

As suggested by [1], SVM is a classical approach and can be considered the most common and useful technique for binary classification tasks. Nevertheless, it is challenging for SVM to find an appropriate kernel to achieve higher accuracy and efficiency in specific tasks. Specifically, for nonlinear cases, there exists no general solution and prediction accuracy cannot be guaranteed. The RF and GBDT methods are an ensemble of DT algorithms and solve over-fitting problems to some extent. However, due to ensembling, the algorithms suffer interpretability and may indicate the classification results to the class with additional samples.

B. Hyperparameter Tuning of Base Classifiers

The simulated annealing (SA) algorithm method for optimizing ML model parameters is preferred for hard computational and practical optimization problems where exact algorithms such as gradient descent have failed [30]. SA is inspired by annealing in metallurgy, which involves the heating and gradual cooling process of the metal to produce defectless crystals. In essence, there are three main steps: initialization, the states transition mechanism and the cooling schedule formulated by an objective function of many variables. Every vector consisting of values of the hyper-parameters can be an element in the population size. The four main steps are executed repeatedly until the optimal values of the parameters given in Table I are obtained:

i. The algorithm starts by randomly initializing the population.

ii. At each iteration, the target is to obtain a better solution in terms of the fitness function.

iii. The probability-based decision decides whether the new solution is preferred or discarded.

iv. At each step, the temperature is progressively decreased from an initial positive value towards zero. A better solution gets a positive moving probability while an inferior solution is assigned zero moving probability.

For parameter tuning, a hyperparameter API is used to automatically configure hyperparameter optimization toolkit [31]. It is highly versatile in model optimization and provides a unified view of possible preprocessing modules and classifiers. Instead of conventional tedious search, it is used to automatically search the best combination of hyperparameters very quickly and can therefore surpass human experts in algorithm configuration.

C. Meta Classifier

In practice, multiple classification models are used for electricity theft detection but none is fully accurate. The stacking of ML methods may improve the performance due to well-performing base models that are skillful on a problem but in a different way [26], [28], [32]. In the multi-model ensemble technique, diverse basic classifiers are trained independently on a given dataset to ensure high parallelism and the predictions of the collection of models at the first stage are provided to the second stage learning (meta classifier) model as an input. The methodology of PFSC is demonstrated in Algorithm 1. The algorithm starts with the data preparation step based on IONB. Three base classifiers ($b_{(1-3)}$) are fitted to the resampled dataset x_i and provide predictions. Each base classifier b_i would give a vector of features which form a new dataset $x_i' = b_1(x_i), b_2(x_i)$ and $b_3(x_i)$. Once the second level classifier is trained, its performance is tested on unseen data.

The main aim of DL based meta classifier development is to detect malicious behavior by targeting the integrity of the readings on consumed energy. For this purpose, different structures of the deep neural network, feedforward, recurrent and convolutional-recurrent neural networks, are investigated to capture complex data representative patterns of energy consumption. Finally, TCN is preferred because of stronger function fitting and better nonlinear computing abilities to learn key features and essential laws from mass

Algorithm 1 PFSC Working for Electricity Theft Detection

Input: Training data $N = \{x_i, y_i\}_{i=1}^{n} \quad (x_i, y_i \in \mathbb{R}^n)$
Output: Obtained results from second-order classifier M
1: Module 1: Data preparation based on IONB
2: Module 2: Learn first-order classifiers
3: **for** $t \leftarrow 1$ to T **do**
4: First-order (base) classifier F_t training on N
5: **end for**
6: Construct a new dataset from D
7: **for** $i \leftarrow 1$ to m **do**
8: Construct a new dataset that comprises $x_i{}' = \{b_1(x_i), b_2(x_i)$ and $b_3(x_i)\}$ from N
9: **end for**
10: Module 3: Learn a second-order classifier
11: Second-order (meta) classifier M training on the newly constructed dataset
12: return $M(x)= m(b_1(x))$, $(m_2(x))$ and $(m_3(x))$.

data. Also, in time-series data analysis tasks, TCN outperforms well-established recurrent networks such as recurrent neural network (RNN) and long short-term memory (LSTM) in terms of accuracy and efficiency [33]. In the following section, we formulate the classification problem and propose its optimization.

D. Problem Formulation

We model the classification problem so as to compute the loss between actual class and predicted class as follows,

$$L = -\frac{1}{N}\left[\sum_{i=1}^{N} y_i - log(h_\theta(x_i)) + (1 - y_i)\ log(1 - h_\theta(x_i))\right] \tag{6}$$

Eq. (6) represents the binary cross entropy loss for N training samples, whilst y_i is the actual class value for the input-output pair (x_i, y_i). To cover the input sequence, the values of hyper-parameters $c_i^{\infty}(i = 1, 2, \ldots,)$ such as kernel size k, dilation factor d and receptive field size r need to be determined. The term $h_\theta(x)$ represents nonlinear hypothesis of convolutional network and can be defined as follows,

$$h_\theta(x) = f\big(w^T x + b\big), \tag{7}$$

where b represents bias and $f(.) : \mathbb{R} \rightarrow \mathbb{R}$ is the activation function. TCN relates to a 1D CNN to encode sequence information [34]. A vanila 1D convolution layer is written as,

$$F(x_t) = (t)(x *_d f) = \sum_{i=0}^{k-1} f(i).X_{s-d.t}, t \geq k \tag{8}$$

where x is the input sequence, $*_d$ is dialated convolutional operator, $f \in R^{k \times d}$ is a convolutional filter with size k, d is dilation coefficient and the term $s-d.t$ represents direction into the past. By stacking several vanila 1D convolutional layers, a 1D CNN is constructed. However, in sequence modeling, 1D CNN is restricted due to limited receptive fields and shrinking output size [34]. By contrast, TCN is featured with causal and dilated convolutional techniques to address these problems.

Causal convolutions: The Module 3 in Fig. 1 shows how a vanila 1D convolutional layer takes n sequences as input and returns $n-k+1$ sequences as output. With more stacked layers, the output sequence shrinkage would increase further.

In time-series data analysis, models are expected to predict for each time step with updates in real-time. This problem is well addressed when a causal convolutional layer allows concatenation of zero paddings of length $k-1$ at the beginning of the input sequence to ensure that the output has the desired length. Due to zero padding, the output tensor makes sure to have the same length as the input tensor. The required number of zero-padding entries p is computed as follows [35],

$$p = b^i.(k - 1) \tag{9}$$

where b is the dilation base and i is the number of layers below the current layer. For a convolutional layer to be causal, the prediction $p(x_{t+1}|x_1, \ldots, x_t)$ only depends on the elements that come before it in the input sequence $\{x_t, x_{t-1}, \ldots, x_{-\infty}\}$ but not on the future indices,

$$F(x_t) = (t)(x *_d f) = \sum_{i=0}^{k-1} f(i).x_{s-d.t}, x_{\leq 0} \coloneqq 0 \tag{10}$$

The causal convolution splits the convolution operation in half so that it can only convolute the information of past time steps. The prediction result of the current state t is only related to historical information, thus avoiding information leakage.

Dilated convolutions: Another disadvantage that pertains to the vanila 1D CNN is its linear receptive nature, which means that the receptive field grows linearly with every additional layer. In long-term dependency modeling such as ETD, the historic data is sufficiently large and the narrow receptive field would cause problems. To circumvent this, dilated convolution enables an exponentially larger receptive field. In the context of a conventional convolutional layer, dilation refers to the gap within the elements of the input sequence that are utilized to calculate one entry of the output sequence. Therefore, a conventional convolutional layer could be regarded as a 1-dilated layer because 1 output value depends on adjacent input elements. Fig. 3 shows the differences between standard, causal, dilated convolutions and zero padding to obtain long-term information. More specifically, receptive field size r of a 1D convolutional network with a kernel size k and n layers can be calculated as,

$$r = 1 + n * (k - 1) \tag{11}$$

whereas for a fixed kernel size k and keeping the receptive field size equal to input length, the required number of layers for full history coverage is calculated as,

$$n = \left[\frac{(l - 1)}{(k - 1)}\right] \tag{12}$$

Eq. (12) states that with a fixed kernel size, the network depth has a direct relationship with the length of the input tensor. For full history coverage, the involvement of a large number of parameters would be required to train the model. Hence, the model would become very deep very quickly and may lead to the degradation of the loss function. One way to increase

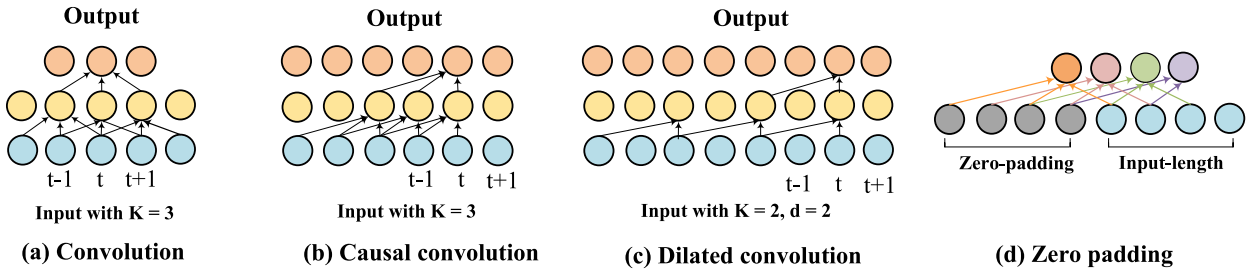


Fig. 3. Differences between (a) Convolution (b) Causal convolutional (c) Dilated causal convolutional and (d) Zero padding.

the receptive field with a relatively small number of layers is to introduce dilation to the convolutional network, as shown in Fig. 3 (c). Fig. 1 (Module 3) also suggests that for full history coverage, the value of the dilation factor d exponentially increases for a specific layer as we move up through the layers. The formulas for exponentially growing the receptive field and dilation are $(k-1)^{r-1}$ and $d = b^i$ respectively. Hence, the width of the receptive field w is computed using Eq. (13) [35],

$$w = 1 + \sum_{i=0}^{n-1}(k-1)\ .\ b^1 = 1 + (k-1)\ .\ \frac{b^n - 1}{b-1} \geq l \quad (13)$$

Without sacrificing receptive field coverage, the dilation factor brings significant improvement in terms of the required number of layers. As opposed to Eq. (12), the minimum number of required layers n for full history coverage are now based on the logarithmic length of the input tensor and dilation base b,

$$n = \left\lceil log_b\left(\frac{(l-1)\ .\ (b-1)}{(k-1)} + 1\right)\right\rceil \quad (14)$$

Fig. 1 (Module 3) shows that a residual block comprises two 1D causal convolutional layers with the same d and k values. The outputs of both layers are added and given to the next residual block as an input. The addition of residual blocks affects the overall requirement of the number of layers and adds twice as much receptive field width for full history coverage. Similarly, regularization techniques such as batch normalization and dropout are introduced after every convolutional layer to prevent overfitting. Finally, the output u of all the temporal convolutional layers is defined as follows,

$$u = (F(x_1), F(x_2), \ldots, F(x_n)) \quad (15)$$

The PFSC performance is more sensitive to the hyperparameters values of TCN, such as kernel size, dilation factor and receptive field size. To determine optimal network configurations, a series of repeated models were generated with different parameter settings and the final prediction accuracy was gauged using the error metrics stated previously. Finally, the tunable parameters of the prediction model using TCN are set as follows: convolution kernel size is 2; number of filters is 64; the dilation factor is set as 2; the learning rate is 0.05; the number of TCN layers is 3; residual connections are adopted between TCN layers; the optimization function of the model is Adam; and the loss function is chosen as binary cross entropy loss.

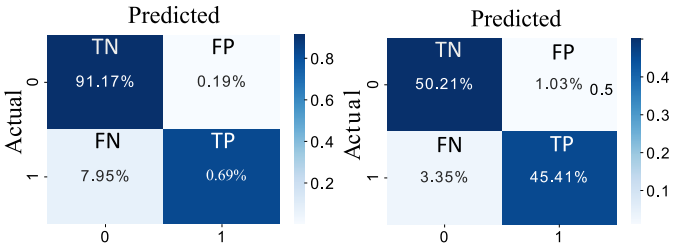


Fig. 4. Confusion matrix before and after resampling.

Based on the integration of IONB, first and second-order classifier adjustment, our framework for ETD can identify the honest and dishonest consumption pattern accurately.

E. Evaluation Metrics

The performance is determined from the CM, i.e., the matrix that is used to explain distinct outcomes in classification problems, as alluded to earlier and shown in Fig. 4. In binary classification tasks, the 0 class label is dedicated for honest consumers and that for dishonest consumers, the class label 1 is assigned [36]. Here, TP (1,1) and TN (0,0) scores mean that normal and abnormal consumption patterns are identified accurately. Similarly, FP (0,1) and FN (1,0) scores mean that the number of customers having normal and abnormal consumption patterns are misclassified. More specifically, FP accounts for those observations in the CM that were honest but predicted dishonest, whilst FN observations contain dishonest consumption patterns that were predicted honest. CM is utilized for the validation of the model's performance in terms of different metrics such as Accuracy, Precision, Recall and the F1 score.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \quad (16)$$

$$Precision = \frac{TP}{TP + FP}, \quad (17)$$

$$Recall = \frac{TP}{TP + FN}, \quad (18)$$

$$F_1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (19)$$

The area under the curve (AUC) represents the degree of separability and provides a more reliable assessment between classes when data distribution has an unequal proportion. It is the probability that a randomly chosen positive sample ranks

TABLE II
METADATA INFORMATION

Description	Value
Electricity consumption time window	01-01-2014 to 31-10-2016
Class of customers	Residential
Power source (conventional, RES)	Utility
Data resolution	Daily data
Total consumers	42372
Honest consumers	38757
Dishonest consumers	3615

higher than a randomly chosen negative sample. For AUC calculations, the formula is as follows [3],

$$AUC = \frac{\Sigma_{i \in PC} Rank_i - \frac{M(1+M)}{2}}{M \times N} \tag{20}$$

where PC is Positive Class, $Rank_i$ is the rank value of sample i in ascending order, M and N represent the number of positive and negative samples. The AUC of receiver operator characteristic (ROC) curve is a graphical demonstration of the false positive rate (FPR) and true positive rate (TPR) plotted on the x-axis and y-axis, respectively. The FPR $\frac{FP}{FP+TN}$ measures the fraction of negative class misclassified as dishonest and TPR, also known as Recall Sensitivity, $\frac{TP}{TP+FN}$ calculates the fraction of positive class labeled correctly. It is pertinent to mention that the range of the ROC lies between 0 and 1. When AUC goes straight up the y axis to approximate 1 and then along the x, it authenticates that the classifier perfectly discriminates both classes. By contrast, if an AUC follows the diagonal line or falls below 0.5, this means that the classifier is randomly guessing and has no power for the classification task.

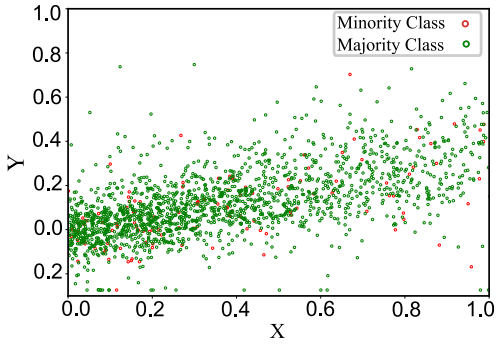
V. EXPERIMENTS AND RESULTS

A. Case Study Setup

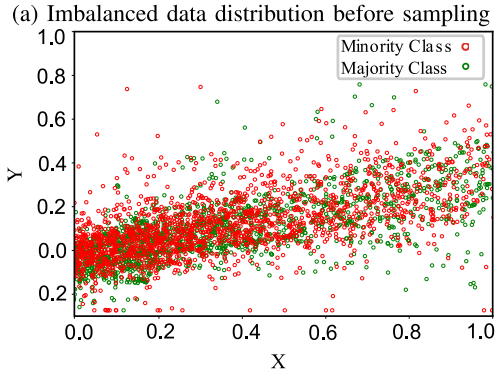
To investigates the capabilities of our proposal, the cases are developed in Google Colaboratory [37] according to the system framework illustrated in Section II. The realistic load profile data is obtained from SGCC [38]. The data contains the electricity consumption record of 42372 users from 2014 to 2016 with a tracked record of 38757 users as fair and the remaining 3615 as fraudster users as shown in Table II.

B. Performance Results:

1) Impact of Handling Imbalanced Class: In an extreme class imbalanced problem, one class predominates the other due to the unequal distribution of classes and thus creates a problem when identifying positive classes. Figs. 5a and 5b show the difference between minority and majority classes before and after handling the class imbalance. Clearly, the majority class customers (green circles) are in a much higher ratio and may cause high bias in the model during the training process. Without dealing with the imbalanced class distribution problem, the CM in Fig. 4a shows severe performance loss and identifies only 0.69%, wherein the reality 9% consumers are fraudulent. The value of FN is 7.95% which means the model has corresponded to the majority class well and considers minority class features as noise to be ignored. The



(a) Imbalanced data distribution before sampling



(b) Balanced data distribution after sampling

Fig. 5. Data distribution.

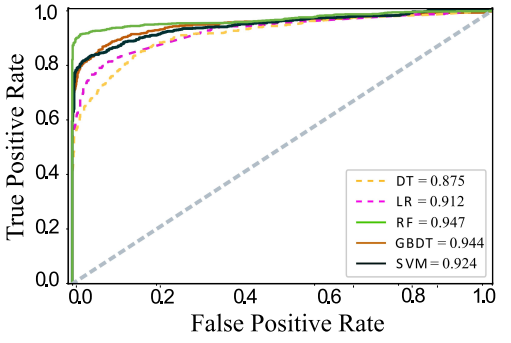


Fig. 6. Standalone base classifiers performance.

model obtained 0.5850 for the AUC score, and 0.7021 and 0.4453 for Precision and Recall performance metrics, respectively. We then apply STLU for balancing the minority and majority classes and the resampled dataset has equal distribution of both classes, i.e., 50% of honest and dishonest customers. After obtaining a balanced distribution for both samples, both model training and model's generalization capabilities are much improved. When the model is applied to the test dataset, the CM in Fig. 5b exhibits that most of the positives and negative cases are correctly identified. The numerical results of each classifier are based on resampled data and achieved the performance metrics shown in Table III.

2) Base Classifiers Performance Comparison With Benchmark Methods: In this case study, five different ML models have been used and, among them, the three best performing models are preferred as first-order classifiers. Fig. 6 shows AUC curves for DT, LR, RF, GBDT and

TABLE III
PERFORMANCE COMPARISON OF INDIVIDUAL BASE AND META CLASSIFIERS

Classifiers	Accuracy	Precision	Recall	F1-Score	AUC	Time (s)
Base classifier						
LR	0.838	0.838	0.838	0.838	0.912	111
DT	0.899	0.899	0.899	0.899	0.894	58
RF	0.874	0.877	0.874	0.874	0.944	37
GBDT	0.884	0.886	0.884	0.884	0.944	85
SVM	0.858	0.868	0.859	0.857	0.924	88
Meta classifiers						
MLP	0.746	0.829	0.749	0.731	0.933	21
LSTM	0.914	0.917	0.914	0.914	0.956	11
GRU	0.944	0.947	0.945	0.944	0.958	15
CNN	0.917	0.918	0.917	0.917	0.978	18
TCN	0.946	0.948	0.946	0.946	0.985	9

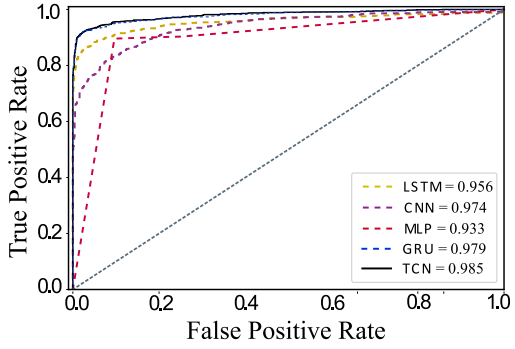


Fig. 7. Standalone meta classifiers performance.

SVM. From the performance curves, it is seen that the RF, GBDT and SVM results are comparable; however DT and LR tend to be weak classifiers for distinguishing honest and dishonest electricity consumptions patterns because of the overfitting problem (and possibly other reasons as discussed in Section IV-A). It is worth noting that the performance of the meta classifier solely depends upon the performance of the base classifiers. Thus, we select RF, GBDT and SVM as base classifiers to guarantee higher accuracy and robustness of final classification and drop DT and LR to avoid overfitting and time complexity problems.

3) Meta Classifier Performance Comparison With Benchmark Methods: In this section, we compare the performance of TCN with other state-of-the-art classifiers such as MLP, LSTM, GRU and CNN. The experimental results for AUC are shown in Fig. 7. Although LSTM and GRU models can achieve improved prediction results, they are still worse than the TCN model, as can be seen from Table III. A notable drawback of LSTM and RNNs is that the sequential structure makes them hard to parallelize since the output for a certain time step depends on the output of previous time steps. The predicted value of the TCN model is nearest to the actual value, which can accurately indicate the dynamic trend of structural deformation. The TCN model effectively increases the receptive field size by stacking the convolutional layer, extending the dilation factor, enlarging the convolution kernel size, and thus better controlling the model's memory length. This evades the gradient explosion problem that often appears in RNNs due to the difference in the back propagation path and sequence time direction [32].

TABLE IV
BENCHMARK FRAMEWORKS

Benchmark	Description
Proposed	IONB + Stacked generalization
E	IONB + DL methods only
D	IONB + ML methods only
C	SMOTE [13]
B	TLU [27]
A	Without sampling

Speed is important and faster networks shorten the feedback cycle. From Table III, it is notable that the computational complexity of the TCN is less than the others for this classification task. This is because massive parallelism shortens both the training and evaluation cycles of TCN. In the meantime, the residual connection can effectively improve the model accuracy. It is also notable that base classifiers require more time for predictions when compared to the meta classifiers. This is because the base classifiers are trained on the original dataset that contains 1035 features with issues like redundancy and irrelevancy. The meta classifiers on the other hand are trained only on the three informative features provided by the base classifiers, hence the conventional problem of computational complexity in DL models has been addressed. The proposed detection architecture achieves an AUC score of 98.5% and an FP of only 1.03%.

4) PFSC Performance on Theft Detection: In this case study, we investigate the capabilities of PFSC and a comparison among different benchmarks is conducted. These benchmarks are given in Table IV. Fig. 8 shows a line plot of accuracy and loss (how good or bad the model's prediction is on a single example) over each epoch. The lower plot shows that the loss is smooth and the training process converges well between the probability distributions. The uneven upper plot for accuracy in Fig. 8 shows that the training and testing sets have binary prediction outcomes with a less granular feedback on performance. The bar plot in Fig. 9 shows that the proposed PFSC framework achieves higher accuracy in ETD than all the benchmarks. The comparison among frameworks A–E suggests that, for these simulation experiments, every module in our proposal can improve the accuracy of the classifier. With the IONB module, the first-order ML classifier gives better results and, finally, the multi-model ensemble method achieves better performance.

TABLE V
COMPARISON AMONG SPRC AND OTHER BENCHMARK SCHEMES

Methods	Training Ratio 60%				Training Ratio 70%				Training Ratio 80%			
	Precision	Recall	F1-Score	AUC	Precision	Recall	F1-Score	AUC	Precision	Recall	F1-Score	AUC
RF	0.550	0.608	0.533	0.694	0.774	0.725	0.725	0.720	0.791	0.744	0.437	0.94
SVM	0.573	0.608	0.534	0.751	0.751	0.753	0.753	0.755	0.774	0.771	0.627	0.92
GRU	0.637	0.614	0.581	0.690	0.688	0.689	0.689	0.690	0.773	0.688	0.787	0.97
CNN	0.748	0.884	0.872	0.811	0.773	0.855	0.855	0.638	0.856	0.865	0.877	0.97
PFSC	0.947	0.912	0.943	0.938	0.961	0.980	0.941	0.938	0.964	0.954	0.959	0.98

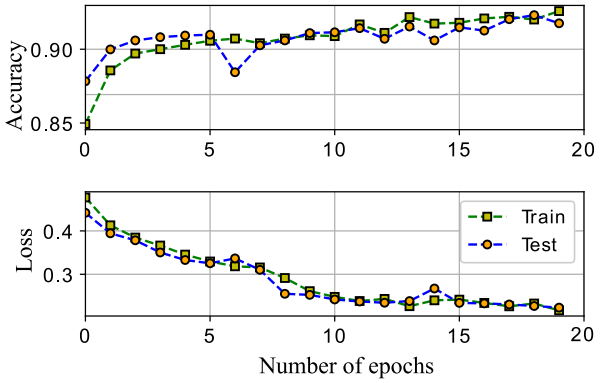


Fig. 8. Performance of PFSC model.

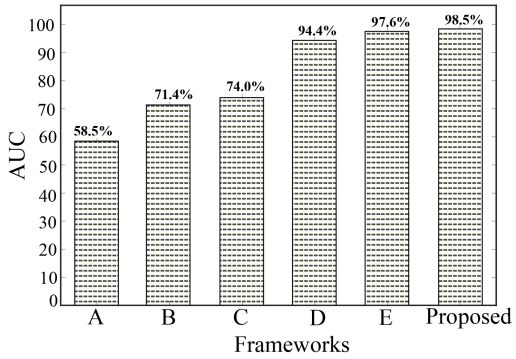


Fig. 9. Comparison of accuracy among PFSC and benchmark frameworks.

5) PFSC Robustness Comparison With Benchmark Algorithms: In a practical setting, ML and DL models are sensitive to various outliers and training data size. In PFSC, we choose stacking, compared to the bagging and boosting methods, since it is more robust to the various outliers for the following two reasons [39]. First, stacking considers heterogeneous weak learners and learns to combine the base models using a meta-model. In contrast, bagging and boosting methods consider homogeneous weak learners following deterministic algorithms.

This case study also intends to affirm whether PFSC maintains its superiority when small, medium and high sizes of training samples (60%, 70% and 80%), compared to the size of all samples, are available for classifier's training. As can be seen from the experimental results provided in Table V, the PFSC outperforms the other algorithms under consideration for all sizes of the training dataset. Results from the conventional schemes show an expanding trend with increased data available. It is observed that PFSC achieves a maximum AUC value of 0.985 and outperforms other algorithms in terms of performance metrics for these data.

VI. CONCLUSION

This paper has proposed a DL-based multi-model ensemble approach, PFSC, to capture abnormal electricity consumption patterns in smart grids. This methodology has been evaluated using realistic electricity consumption data issued by SGCC, the largest power utility in China. The obtained results have shown that with the proposed ensemble method, the complex relationships among the classifiers are determined automatically and efficiently, thus allowing the ensemble approach to improve the performance of the prediction model. The method has attained an AUC score of 0.985 on the real dataset. The DL-based multi-model ensemble approach minimizes the generation error and captures valuable information by employing the first-stage predictions as input features. These results show that the proposed IONB and stacked generalization method outperform both base ML and meta DL approaches. Moreover, the comparison with other state-of-the-art classifiers has proved that the proposed ensemble model can exceed the performance of those established classifiers such as SVM, RF, GBDT, ANN, CNN, LSTM and GRU in terms of accuracy and robustness, and thus can effectually be utilized in industrial applications.

In the future, the PFSC performance may be improved with two further investigations. First, knowledge from grid sources, network distribution topology, class of customers, seasonalities and geographic information, will be exploited for monitoring abnormalities in energy consumption patterns. Second, the robustness of the proposed method will be demonstrated with synthetically generated theft attacks and adding random noise (error) in selected data to observe the average accuracy of base and meta classifiers.

REFERENCES

[1] K. Wang, C. Xu, Y. Zhang, S. Guo, and A. Y. Zomaya, "Robust big data analytics for electricity price forecasting in the smart grid," *IEEE Trans. Big Data*, vol. 5, no. 1, pp. 34–45, Mar. 2019.

[2] P. P. Biswas, H. Cai, B. Zhou, B. Chen, D. Mashima, and V. W. Zheng, "Electricity theft pinpointing through correlation analysis of master and individual meter readings," *IEEE Trans. Smart Grid*, vol. 11, no. 4, pp. 3031–3042, Jul. 2020.

[3] Z. Zheng, Y. Yang, X. Niu, H.-N. Dai, and Y. Zhou, "Wide and deep convolutional neural networks for electricity-theft detection to secure smart grids," *IEEE Trans. Ind. Informat.*, vol. 14, no. 4, pp. 1606–1615, Apr. 2018.

[4] T. Hu, Q. Guo, X. Shen, H. Sun, R. Wu, and H. Xi, "Utilizing unlabeled data to detect electricity fraud in AMI: A semisupervised deep learning approach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 11, pp. 3287–3299, Nov. 2019.

[5] W. Li, T. Logenthiran, V.-T. Phan, and W. L. Woo, "A novel smart energy theft system (SETS) for IoT-based smart home," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 5531–5539, Jun. 2019.

[6] K. Wang, H. Li, Y. Feng, and G. Tian, "Big data analytics for system stability evaluation strategy in the energy Internet," *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 1969–1978, Aug. 2017.

[7] K. Wang, Z. Ouyang, R. Krishnan, L. Shu, and L. He, "A game theory-based energy management system using price elasticity for smart grids," *IEEE Trans. Ind. Informat.*, vol. 11, no. 6, pp. 1607–1616, Dec. 2015.

[8] M. M. Buzau, J. Tejedor-Aguilera, P. Cruz-Romero, and A. Gómez-Expósito, "Detection of non-technical losses using smart meter data and supervised learning," *IEEE Trans. Smart Grid*, vol. 10, no. 3, pp. 2661–2670, May 2019.

[9] 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 Trans. Ind. Informat.*, vol. 12, no. 3, pp. 1005–1016, Jun. 2016.

[10] A. M. Tureczek and P. S. Nielsen, "Structured literature review of electricity consumption classification using smart meter data," *Energies*, vol. 10, no. 5, p. 584, 2017.

[11] Z. Aslam, N. Javaid, A. Ahmad, A. Ahmed, and S. M. Gulfam, "A combined deep learning and ensemble learning methodology to avoid electricity theft in smart grids," *Energies*, vol. 13, no. 21, p. 5599, 2020.

[12] S. Li, Y. Han, X. Yao, S. Yingchen, J. Wang, and Q. Zhao, "Electricity theft detection in power grids with deep learning and random forests," *J. Elect. Comput. Eng.*, vol. 2019, Oct. 2019, Art. no. 4136874.

[13] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jan. 2002.

[14] P. Jokar, N. Arianpoo, and V. C. M. Leung, "Electricity theft detection in AMI using customers' consumption patterns," *IEEE Trans. Smart Grid*, vol. 7, no. 1, pp. 216–226, Jan. 2016.

[15] N. Javaid, N. Jan, and M. U. Javed, "An adaptive synthesis to handle imbalanced big data with deep siamese network for electricity theft detection in smart grids," *J. Parallel Distrib. Comput.*, vol. 153, pp. 44–52, Jul. 2021.

[16] J. A. Meira *et al.*, "Distilling provider-independent data for general detection of non-technical losses," in *Proc. IEEE Power Energy Conf. Illinois (PECI)*, Champaign, IL, USA, 2017, pp. 1–5.

[17] H. Han, J. Dang, and E. Ren, "Comparative study of two uncertain support vector machines," in *Proc. IEEE 5th Int. Conf. Adv. Comput. Intell. (ICACI)*, Nanjing, China, 2012, pp. 388–390.

[18] D. Saxena and J. Cao, "Generative adversarial networks (GANs): Challenges, solutions, and future directions," *ACM Comput. Surv.*, vol. 54, no. 3, pp. 1–42, 2021.

[19] Y. Xiao, J. Wu, Z. Lin, and X. Zhao, "A deep learning-based multi-model ensemble method for cancer prediction," *Comput. Methods Programs Biomed.*, vol. 153, pp. 1–9, Jan. 2018.

[20] H. Liu and C. Chen, "Multi-objective data-ensemble wind speed forecasting model with stacked sparse autoencoder and adaptive decomposition-based error correction," *Appl. Energy*, vol. 254, Nov. 2019, Art. no. 113686.

[21] J. Xu, W. Wang, H. Wang, and J. Guo, "Multi-model ensemble with rich spatial information for object detection," *Pattern Recognit.*, vol. 99, Mar. 2020, Art. no. 107098.

[22] A. Khamparia *et al.*, "A novel deep learning-based multi-model ensemble method for the prediction of neuromuscular disorders," *Neural Comput. Appl.*, vol. 32, no. 15, pp. 11083–11095, 2020.

[23] I. U. Khan, N. Javaid, C. J. Taylor, K. A. Gamage, and X. Ma, "Big data analytics for electricity theft detection in smart grids," in *Proc. IEEE Madrid PowerTech*, Madrid, Spain, 2021, pp. 1–6.

[24] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 1–58, 2009.

[25] N. F. Avila, G. Figueroa, and C.-C. Chu, "NTL detection in electric distribution systems using the maximal overlap discrete wavelet-packet transform and random undersampling boosting," *IEEE Trans. Power Syst.*, vol. 33, no. 6, pp. 7171–7180, Nov. 2018.

[26] J. Brownlee. "How to Combine Oversampling and Undersampling for Imbalanced Classification." Jan. 2020. [Online]. Available: https://machinelearningmastery.com/combine-oversampling-and-undersampling-for-imbalanced-classification/

[27] G. E. Batista, A. L. Bazzan, and M. C. Monard, "Balancing training data for automated annotation of keywords: A case study," in *Proc. WOB*, 2003, pp. 10–18.

[28] S. Džeroski and B. Ženko, "Is combining classifiers with stacking better than selecting the best one?" *Mach. Learn.*, vol. 54, no. 3, pp. 255–273, 2004.

[29] J. Tang, S. Alelyani, and H. Liu, *Data Classification: Algorithms and Applications* (Data Mining and Knowledge Discovery Series). Boca Raton, FL, USA: CRC Press, 2014, pp. 37–64.

[30] D. Oliva, M. Abd Elaziz, A. H. Elsheikh, and A. A. Ewees, "A review on meta-heuristics methods for estimating parameters of solar cells," *J. Power Sources*, vol. 435, Sep. 2019, Art. no. 126683.

[31] S. Blanke. "An Optimization and Data Collection Toolbox for Convenient and Fast Prototyping of Computationally Expensive Models." [Online]. Available: https://github.com/SimonBlanke/Hyperactive (Accessed: Oct. 7, 2021).

[32] J. Brownlee. "Stacking Ensemble for Deep Learning Neural Networks in Python." Dec. 2018. [Online]. Available: https://machinelearningmastery.com/stacking-ensemble-for-deep-learning-neural-networks/

[33] G. De Giacomo, A. Catala, and B. Dilkina, Eds., *ECAI 2020: 24th European Conference on Artificial Intelligence, 29 August-8 September 2020, Santiago de Compostela, Spain-Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020)*, vol. 325. Amsterdam, The Netherlands: IOS Press, 2020.

[34] J. You, Y. Wang, A. Pal, P. Eksombatchai, C. Rosenburg, and J. Leskovec, "Hierarchical temporal convolutional networks for dynamic recommender systems," in *Proc. World Wide Web Conf.*, 2019, pp. 2236–2246.

[35] F. Lässig. "Temporal Convolutional Networks and Forecasting." Oct. 2020. [Online]. Available: https://medium.com/unit8-machine-learning-publication/temporal-convolutional-networks-and-forecasting-5ce1b6e97ce4

[36] A. Arif, N. Javaid, A. Aldegheishem, and N. Alrajeh, "Big data analytics for identifying electricity theft using machine learning approaches in microgrids for smart communities," *Concurrency Comput. Pract. Exp.*, vol. 33, no. 17, p. e6316, 2021.

[37] E. Bisong, *Building Machine Learning and Deep Learning Models on Google Cloud Platform*. Berkeley, CA, USA: Springer, 2019.

[38] "Electricity Theft Detection." 2018. [Online]. Available: https://github.com/henryRDlab/ElectricityTheftDetection

[39] J. Rocca. "Ensemble Methods: Bagging, Boosting and Stacking." [Online]. Available: https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205 (Accessed: Sep. 7, 2021).