A Robust Hybrid Deep Learning Model for Detection of Non-Technical Losses to Secure Smart Grids

FAISAL SHEHZAD¹, NADEEM JAVAID¹, (Senior Member, IEEE),
AHMAD ALMOGREN², (Senior Member, IEEE), ABRAR AHMED³,
SARDAR MUHAMMAD GULFAM³, AND AYMAN RADWAN⁴, (Senior Member, IEEE)

¹Department of Computer Science, COMSATS University Islamabad, Islamabad 44000, Pakistan
²Department of Computer Science, College of Computer and Information Sciences, King Saud University, Riyadh 11633, Saudi Arabia
³Department of Electrical and Computer Engineering, COMSATS University Islamabad, Islamabad 44000, Pakistan
⁴Instituto de Telecomunicacoes, Universidade de Aveiro, 3810-193 Aveiro, Portugal

Corresponding authors: Nadeem Javaid (nadeemjavaidqau@gmail.com) and Ahmad Almogren (ahalmogren@ksu.edu.sa)

This work was supported by King Saud University, Riyadh, Saudi Arabia, through the Researchers Supporting Project under Grant RSP-2021/184. The work of Ayman Radwan was supported in part by FCT/MEC through the Programa Operacional Regional do Centro, and in part by European Union through European Social Fund (ESF) under Investigador FCT Grant 5G-AHEAD IF/FCT-IF/01393/2015/CP1310/CT0002.

ABSTRACT For dealing with the electricity theft detection in the smart grids, this article introduces a hybrid deep learning model. The model tackles various issues such as class imbalance problem, curse of dimensionality and low theft detection rate of the existing models. The model integrates the benefits of both GoogLeNet and gated recurrent unit (GRU). The one dimensional electricity consumption (EC) data is fed into GRU to remember the periodic patterns of electricity consumption. Whereas, GoogLeNet model is leveraged to extract the latent features from the two dimensional weekly stacked EC data. Furthermore, the time least square generative adversarial network (TLSGAN) is proposed to solve the class imbalance problem. The TLSGAN uses unsupervised and supervised loss functions to generate fake theft samples, which have high resemblance with real world theft samples. The standard generative adversarial network only updates the weights of those points that are available at the wrong side of the decision boundary. Whereas, TLSGAN even modifies the weights of those points that are available at the correct side of decision boundary that prevent the model from vanishing gradient problem. Moreover, dropout and batch normalization layers are utilized to enhance model’s convergence speed and generalization ability. The proposed model is compared with different state-of-the-art classifiers including multilayer perceptron (MLP), support vector machine, naive bayes, logistic regression, MLP-long short term memory network and wide and deep convolutional neural network. It outperforms all classifiers by achieving 96% and 97% precision-recall area under the curve and receiver operating characteristics area under the curve, respectively.

INDEX TERMS Electricity theft detection, gated recurrent unit, GoogLeNet, non-technical losses, smart grids, SGCC.

I. INTRODUCTION

Two types of losses occur during generation, transmission and distribution of electricity that are technical and non-technical losses (NTLs). The latter are caused by meter tampering, direct hooking to transmission lines, billing errors, faulty meters, etc. These losses not only affect the performance of electricity generation companies, however, they also damage their physical components. Moreover, a recent report shows that NTLs cause $96 billion of revenue loss every year [1]. According to the World Bank’s report, India, China and Brazil bear 25%, 6% and 16% loss on their total electric supply, respectively. The NTLs are not limited to only developing countries; it is estimated that developed countries like UK and USA also lose 232 million and 6 billion US dollars per annum, respectively [2].

Electricity theft is a primary cause of NTLs. The evolution of advanced metering infrastructure (AMI) promises to
TABLE 1. List of abbreviations.

| Abbreviation | Full form |
|--------------|-----------|
| ADASYN       | Adaptive synthetic sampling approach |
| AMI          | Advanced metering infrastructure |
| CNN          | Convolutional neural network |
| CPPETD       | Consumption pattern based electricity theft detector |
| CBoost       | Categorical boosting |
| D            | Discriminator |
| DR           | Detection rate |
| EC           | Electricity consumption |
| EDT          | Electricity theft detection |
| FPR          | False positive rate |
| GKR          | Guided recurrent unit |
| G            | Generator |
| KNN          | k-nearest neighbors |
| LSTM         | Long short term memory |
| LR           | Logistic regression |
| LSGAN        | Least square generative adversarial network |
| LightGBM     | Light gradient boosting machine |
| ML           | Machine learning |
| SVM          | Support vector machine |
| SGCC         | Smart grid corporation of China |
| SMOTE        | Synthetic minority over-sampling technique |
| SMOTE_EENN   | SMOTE and edited nearest neighbors |
| MLP          | Multilayer perceptron |
| NTLs         | Non-technical losses |
| NB           | Naive bayes |
| NaN          | Not a number |
| TLSGAN       | Time LSGAN |
| TPR          | True positive rate |
| TSR          | Three sigma rule |
| SSDAE        | Stacked sparse denoising autoencoder |
| RUS          | Random undersampling |
| RF           | Random forest |
| ROS          | Random oversampling |
| RNN          | Recurrent neural network |
| RDNN         | Real-valued deep belief network |
| PR-AUC       | Precision recall area under curve |
| PCA          | Principal component analysis |
| ROC-AUC      | Receiver operating characteristic - area under curve |
| Tls          | Technical losses |
| WDCNN        | Wide and deep convolutional neural network |
| XGBoost      | xGradient boosting |
| $a$          | Label of theft sample |
| $b$          | Label of fake sample |
| $b_{HG2}$    | Bias of hybrid layer |
| $\delta_{HG2}$ | Distance variable |
| $Dense_{GoogLeNet}$ | Last layer of GoogLeNet |
| $Dense_{GRU}$ | Last layer of GRU |
| $E$          | Expected value of all instances |
| $h_t$        | Hidden state at timestamp $t$ |
| $h_{HG2}$    | Hidden layer of hybrid module |
| $h$          | Candidate value |
| $P_{data}(e)$ | Theft data |
| $P_{\Theta}(e)$ | Gaussian distribution |
| $r$          | Reset gate |
| $w_{1}$      | 1st week EC |
| $w_{2}$      | 2nd week EC |
| $W_r$        | Weight of reset gate |
| $W_u$        | Weight of update gate |
| $W_{HG2}$    | Weight of hybrid layer |
| $x_i$        | Complete consumption history of consumer $i$ |
| $x_{ij}$     | Daily EC of a consumer $i$ over time period $j$ (a day) |
| $x_{ji}$     | EC of a previous day |
| $x_{n+j}$    | EC of a next day |
| $\bar{x}_i$  | Average consumption of consumer $i$ |
| $\sigma(x_i)$ | Standard deviation of consumer $i$ |
| $\text{min}(x_i)$ | Minimum value of consumer $i$ |
| $Y_{NTLs}$   | Output of having NTLs or not |
| $z$          | Update gate |

overcome electricity theft through monitoring users’ consumption history. However, it introduces new types of cyber-attacks, which are difficult to detect using conventional methods. Whereas, traditional meters are only compromised through physical tampering. In AMI, the meter readings are tampered locally and remotely over the communication links before sending them to an electric utility [3]. There are three types of approaches to address the NTLs in AMI: state, game theory and data-driven. State-based approaches exploit wireless sensors and radio frequency identification tags to detect NTLs. However, these approaches require high installation, maintenance and training cost and they also perform poorly in extreme weather conditions [4], [5]. Beside this, game theory based approaches hold a game between a power utility and consumers to achieve equilibrium state and then extract hidden patterns from users’ EC history. However, it is difficult to design a suitable utility function for utilities, regulators, distributors and energy thieves to achieve equilibrium state within the defined time [6]. Moreover, both NTLs detection approaches have low detection rate (DR) and high false positive rate (FPR).

The data driven methods get high attention due to the availability of electricity consumption (EC) data that is collected through AMI. A normal consumer’s EC follows a statistical pattern, whereas, abnormal EC does not follow any pattern. The machine learning (ML) and data mining techniques are trained on collected data to learn normal and abnormal consumption patterns. After training, the model is deployed in a smart grid to classify incoming consumer’s data into normal or abnormal samples. Since, these techniques use already available data and do not require to deploy hardware devices at consumers’ site that is why their installation and maintenance costs are low as compared to hardware based methods. However, class imbalance problem is a serious issue for data driven methods where the number of normal EC samples is more than theft ones. Normal data is easily collected through users’ consumption history.

Whereas, theft cases are relatively rare than normal class in the real world that is why few number of samples are present in user’s consumption history. So, little amount of theft samples affect the performance of ML and DL models. They become biased towards the class, which has a large number of samples and ignore other classes [7], [8]. In literature, the authors mostly use random undersampling (RUS) and random oversampling (ROS) techniques to handle the class imbalance problem. However, both techniques have underfitting and overfitting issues that increase the FPR and minimize the DR [3], [9]–[11]. The second challenging issue is the curse of dimensionality. A time series dataset contains a large number of timestamps (features) that increase both execution time and memory complexity and reduce the generalization

\footnote{Theft and abnormal words are used interchangeably.}

\footnote{Benign and normal words are used interchangeably.}
The main contributions of this research article are given below:

- A state of art methodology is proposed that is based on GRU and GoogLeNet. The automatic feature learning mechanism of both models increases convergence speed, accuracy and handles the curse of dimensionality. Moreover, this study integrates the benefits of both 1D and 2D EC data in a parallel manner.
- The TLSGAN is proposed to generate fake samples from existing theft patterns to tackle the class imbalance ratio.
- GRU model is utilized to handle non-malicious factors like sudden changes in EC patterns due to increase in family members, change in weather conditions, etc., and
- Extensive experiments are conducted on a EC data that is collected by smart grid corporation of China (SGCC), the largest smart grid company in China. Different performance indicators are utilized to check the performance of the HG\(^2\) model.

The remaining paper is organized as follows. Sections II and III describe the related work and problem statement, respectively. Section IV illustrates the data preprocessing steps while Section V presents the working mechanism of TLSGAN for solving class imbalance problem. The description of proposed model and experimental analysis are presented in Sections V-B and Section VI, respectively. Finally, the research article is concluded in Section VII.

II. RELATED WORK

In this Section, we discuss limitations of existing literature work. In [3], the authors extend existing consumption pattern based electricity theft detector (CPBETD) that is based on support vector machine (SVM) to detect the abnormal patterns from EC data. However, no feature engineering technique is applied to select optimal features from EC data. The high dimensionality of data creates time complexity, storage and FPR issues. In [7], [10], [15]–[17], feature selection is an important part of data-driven techniques where significant features are selected from existing ones. During feature selection process, less domain knowledge increases FPR and decreases classification accuracy. In [9], previous studies use only an EC dataset to train ML classifiers and predict abnormal patterns. They do not use smart meter data and auxiliary data (geographical information, meter inside or outside, etc.) to predict abnormal patterns from electricity data. In [18], [19], there are various consumption behaviors of different users. The consumption behavior of each customer gives different results. So, it is necessary to select those features, which give best results. However, consumption behaviours are closely related and significant correlation exists between these features. The authors remove highly correlated and overlapped features, which helps to improve DR and decrease FPR. In [11], [20], [21], the authors work on feature engineering and describe why the curse of dimensionality affects the performance of ML models. In [9], the authors generate new features from the smart meter and auxiliary data. These features are based on z-score, electrical magnitude, users’ consumption patterns through clustering technique, smart meter alarm system, geographical location and smart meter’s placement. In [22], features are selected from existing features based on clustering evaluation criteria. In [8], the authors propose a new deep learning model, which has ability to learn and extract latent features from EC data. In [14], the authors use the black hole algorithm to select the optimal number of features and compare the results with particle swarm optimization, differential evolution, genetic algorithm and harmony search. In [20], the authors perform work on feature engineering and identify different features like electricity contract, geographical location, weather condition, etc. In [16], conventional methods are applied on data to tackle the curse of dimensionality issue. This process is very tedious and time-consuming.

In [17], one of the main contributions of this paper is to find optimal number features. It is observed that not all features equally contribute to prediction results. In [15], the authors...
use Dense-Net based convolutional neural network (CNN) to analyse periodicity in EC data. The convolutional layers can capture the long-term and short-term sequences from weekly and monthly EC patterns. In [11], maximal overlap discrete wavelet packet transform is leveraged to extract the optimal features. In [21], the authors implement a bidirectional Wasserstein GAN to extract the optimal features from time series data. In [9], the authors pass a combination of newly created features in different conventional ML classifiers and compare their results. In [18], the authors perform comparison between a number of selected features and classification accuracy. In [8], [23], the authors measure precision and recall score of long short term memory (LSTM) classifier on test data. The hybrid of multilayer perceptron (MLP) and LSTM outperform the single LSTM in terms of PR curve because MLP adds additional information to the network like meter location, contractual data and technical information.

In [20], the identified features are passed to gradient boosting classifiers to classify between normal and abnormal samples. In [2], [9], [24], the authors do not use any feature engineering technique to extract or select the optimal features from high dimensional time series dataset. The high dimensionality of data creates time complexity, storage issues and affects the model generalization ability. In [18], the authors form a feature library where they select a subset of features from existing features using clustering evaluation criteria. However, they do not compare the adopted feature selection strategy with other feature selection strategies. In [2], [3], [10], [18], [25], data imbalance is a major issue for training of ML classifiers. Benign samples are easily collected by getting the history of any consumers. Whereas, theft cases rarely happen in the real world. So, lack of theft samples limit classification accuracy and increase FPR. Generally, there are RUS and ROS techniques are utilized to solve data imbalance problem. In [26], Chawla et al. propose synthetic minority oversampling technique (SMOTE) to create artificial samples of minority class. It has many advanced versions like Random-SMOTE, Kmeans-SMOTE, etc. However, these sampling techniques do not represent the overall distribution of data, which affects the model performance. In [2], the authors introduce six theft cases to generate malicious samples using benign samples. They argue that goal of theft is to report less consumption than actual consumption or shift load toward low tariff periods. After generating malicious samples, the authors exploit ROS technique to solve class imbalance problem.

In [2], the authors argue that the goals of thieves are to report less meter readings, shift load from on-peak hours to off-peak hours, etc. So, it is possible to generate theft samples from normal samples. In [18], [19], the authors use 1D-Wasserstein GAN and adaptive synthetic (ADASYN) to generate duplicated copies of the minority class. In [2], [3], [10], SMOTE and ROS are leveraged to solve the uneven distribution of samples. In [7], [8], [15], [17], [20], [21], [27], [28], the authors do not tackle the above mentioned problems. In [25], the authors use SMOTE and near miss method to handle class imbalance problem. In [9], [11], the authors do not compare the quality of data is improved by applying different techniques that directly enhance the performance of ML methods. The data preprocessing phase contains the following steps, which are explained below.

III. PROBLEM STATEMENT

In [2], the authors propose a CPBETD to identify normal and abnormal EC patterns. However, the CPBETD does not use any feature engineering technique to solve the curse of dimensionality issue. This issue refers to a set of problems that occurs due to high dimensionality of a dataset. A dataset, which contains a large number of features, generally in order of hundreds or more, is known as a high dimensional dataset. A time series dataset has high dimensionality that increases time complexity, reduces DR and affects the generalization of a classifier. In [7], [8], the authors solve the curse of dimensionality issue by selecting the prominent features through deep learning and meta-heuristic techniques. However, the authors do not address class imbalance problem, which is a major issue in NTLs detection. In [3], [25], the authors use SMOTE to handle class imbalance ratio. However, SMOTE creates an overfitting problem. It does not perform well on time series data. In [9], the authors use RUS technique to handle class imbalance ratio. However, this approach discards the useful information from data, which creates an underfitting issue.

IV. DATA PREPROCESSING

Data preprocessing is an important part of data science where the quality of data is improved by applying different techniques that directly enhance the performance of ML methods. The data preprocessing phase contains the following steps, which are explained below.

A. ACQUIRING THE DATASET

The performance of the proposed model is evaluated through SGCC dataset. It contains consumers’ IDs, daily EC and labels either 0 or 1. It comprises EC data of 42,372 consumers, out of which 91.46% are normal and remaining are thieves. Each consumer is labeled as either 0 or 1, where 0 represents normal consumer and 1 represents electricity thief. These labels are assigned by SGCC after performing on-site inspections. The dataset is in a tabular form. The rows represent complete record of each consumer. While columns represent daily EC of all consumers. The meta information about dataset is given in Table 2.

| Time window                  | Jan. 1, 2014 to Oct. 31, 2016 |
|------------------------------|--------------------------------|
| Total consumers              | 42372                         |
| Normal consumers             | 38757                         |
| Electricity thieves           | 3615                          |

TABLE 2. Dataset information.
### B. HANDLING THE MISSING VALUES

EC datasets often contain missing or erroneous values, which are presented as not a number (NaN). The values often occur due to many reasons: failure of smart meter, fault in distribution lines, unscheduled maintenance of a system, data storage problem, etc. Training data with missing values have negative impact on the performance of ML methods. One way to handle the missing values is to remove the consumers’ records that have missing values. However, this approach may remove valuable information from data. In this study, we use a linear imputation method to recover missing values [3].

\[
    f(x_i) = \begin{cases} 
    \frac{x_{i,j-1} + x_{i,j+1}}{2}, & x_{i,j} = \text{NaN}, x_{i,j+1} \neq \text{NaN}, \\
    0, & x_{i,j-1} = \text{NaN} \text{ or } x_{i,j+1} = \text{NaN}, \\
    x_{i,j}, & x_{i,j} \neq \text{NaN}.
    \end{cases}
\] (1)

In Equation (1), \(x_{i,j}\) represents daily EC of a consumer \(i\) over time period \(j\) (a day). \(x_{i,j-1}\) represents EC of the previous day. \(x_{i,j+1}\) represents the EC of the next day.

#### Algorithm 1 Data Preprocessing Steps

1. **Data:** EC dataset: \(X\)
2. **Variables:** \(\text{min}_i\) = minimum value of consumer \(i\), \(\text{max}_i\) = maximum value of consumer \(i\), \(\bar{x}_i\) = mean of consumer \(i\), \(\sigma_i\) = standard deviation of consumer \(i\), \(row, col = X\_shape\)
3. for \(i \leftarrow \text{row}\) do
   4. for \(j \leftarrow \text{col}\) do
      5. Fill missing values:
         6. if \(x_{i,j-1} \&\& x_{i,j+1} \neq \text{NaN} \&\& x_{i,j} == \text{NaN}\) then
            7. \(x_{i,j} = (x_{i,j-1} + x_{i,j+1})/2\)
         8. end
      9. if \(x_{i,j-1} \parallel x_{i,j+1} == \text{NaN} \text{ then}\)
        10. \(x_{i,j} = 0\)
     11. end
   12. Remove outliers:
       13. if \(x_{i,j} > \bar{x}_i + 3\sigma_i\) then
           14. \(x_{i,j} = \bar{x}_i + 3\sigma_i\)
       15. end
   16. Min-max normalization:
       17. \(x_{i,j} = \frac{x_{i,j} - \text{min}_i}{\text{max}_i - \text{min}_i}\)
   18. end
   19. Result: \(X_{normalized} = X\)

### C. REMOVING THE OUTLIERS FROM DATASET

We have found some outliers in the EC dataset. One of the most important steps of data preprocessing phase is to detect and treat outliers. The supervised learning models are sensitive to the statistical distribution of data. The outliers mislead the training process as a result the models take longer time for training and generate false results. Motivated from [7], three-sigma rule (TSR) is leveraged to remove outliers. The mathematical form of TSR is given in Equation (2).

\[
    f(x_i) = \begin{cases} 
    \bar{x}_i + 3 \times \sigma(x_i), & \text{if } x_{ij} > \bar{x}_i + 3 \times \sigma(x_i), \\
    x_{ij}, & \text{otherwise}.
    \end{cases}
\] (2)

\(x_i\) represents complete energy EC history of consumer \(i\). The \(\bar{x}_i\) denotes average EC and \(\sigma(x_i)\) represents standard deviation of consumer \(i\).

### D. NORMALIZATION

After handling the missing values and outliers, min-max technique is applied to normalize the dataset values because all deep learning models are sensitive to the diversity of data [7]. The experimental results show that deep learning models give good results on normalized data. The mathematical form of min-max technique is given in equation (3).

\[
    x_{ij} = \frac{x_{ij} - \text{min}(x_i)}{\text{max}(x_i) - \text{min}(x_i)}
\] (3)

The \(\text{min}(x_i)\) and \(\text{max}(x_i)\) represent minimum and maximum values of EC of consumer \(i\), respectively. All data preprocessing steps are shown in algorithm 1. In line number 1 and 2, the dataset is acquired from an electric utility and variables are initialized. In line number 3 to 19, following steps are performed: remove missing values, handle outliers and apply the min-max normalization technique. Finally, we obtain a normalized dataset.

### E. EXPLORATORY DATASET ANALYSIS

Electricity theft is a criminal behavior, which is done by tampering or bypassing smart meters, hacking smart meters through cyber attacks and manipulating meter readings using physical components or over the communication links. Since EC data contains normal and abnormal patterns, that is why data driven approaches receive high attention from research community to differentiate between benign and thief consumers. We conduct a preliminary analysis on EC data through statistical techniques to check existence of periodicity and non-periodicity in consumers’ EC patterns. Meta information about dataset is given in Section IV-A. Figure 1a shows the EC pattern of a normal consumer during a month. There are a lot of fluctuations in a monthly EC pattern. So, it is difficult to find normal and abnormal patterns from 1D time series data. Figure 1b shows EC patterns of a normal consumer according to weeks. The EC is decreasing on days 3 and 5, whereas, it is increasing on days 2 and 4. While, 2nd week shows abnormal pattern, which is different from other weeks. We also conduct similar type of analysis on theft patterns. Figures 1c and 1d show EC during a month and a week of an energy thief. There are a lot of fluctuations in monthly measurements and no periodicity exists in weekly EC patterns.

Moreover, the correlation analysis is conducted between EC of thieves and normal consumers. Figure 1e shows
F. Shehzad et al.: Robust Hybrid Deep Learning Model for Detection of NTLs to Secure Smart Grids

FIGURE 1. Statistical analysis between normal and abnormal EC.

Pearson correlation values of a normal consumer that are mostly more than 0.3. It is the indication of a strong relationship between weekly EC patterns of a normal consumer. Figure 1f shows Pearson correlation values of electricity thief, which indicate poor correlation between weekly EC data. Hereinafter, we use Euclidean distance similarity measure to examine how much weekly observations are similar to each other. Euclidean distance is calculated for both normal and theft consumers. We compare EC pattern of the last week of a month with the previous three weeks and then take the average of differences to decide how much normal EC is different to abnormal EC. We observe that the Euclidean distance between normal EC pattern is low as compared to abnormal ones. Similar type of findings are found in the whole dataset. To avoid the repetition, exploratory data analysis is conducted on some observations, which are shown in Figure 1 and Table 3.

\[ f(x) = \sqrt{(w_{ij} - w_{mj})^2 + \ldots + (w_{i,j-n} - w_{m,j-n})^2}. \]  (4)

Equation (4) shows Euclidean distance formula to measure similarity between weekly EC pattern. The \( w_i \) and \( w_m \) denote \( i \)th and \( m \)th weeks. \( j \) is a EC of a specific week day \( j \leq 5 \).

After conducting statistical analysis on thieves and normal consumers, we conclude that theft patterns have more fluctuations (less periodic) than normal EC patterns. We believe that this type of patterns can also be observed in datasets, which are collected from different regions of countries.

TABLE 3. Euclidean distance similarity measure.

|         | \( w_1, w_2 \) | \( w_1, w_3 \) | \( w_1, w_4 \) | Average |
|---------|----------------|----------------|----------------|---------|
| Normal  | 4.70           | 4.83           | 3.66           | 4.40    |
| Theft   | 4.66           | 3.54           | 1.90           | 7.05    |

However, it is challenging to capture long-term periodicity from 1D time series dataset because it consists of long sequential patterns. The conventional statistical and ML models, such as autoregressive integrated moving average, SVM and decision tree are unable to retrieve these patterns. Based on the above analysis, we pass 1D data to GRU model because it is specially designed to capture temporal patterns from time series data. Whereas, 1D EC data is stacked according to weeks and is fed into GoogLeNet to extract periodicity between weeks.

V. THE PROPOSED MODEL

The proposed system model contains following steps:

- handling the class imbalance problem using TLGAN,
- extracting prominent features utilizing GRU and GoogLeNet,
- classifying the theft and benign samples leveraging fully connected neural network,
- handling the non-malicious factors using memory units of GRU and
- enhancing the model’s generalization ability with the help of dropout and batch normalization layers.

Each of the above mentioned steps is explained in the following subsections.

A. HANDLING THE CLASS IMBALANCE PROBLEM

One of the critical problems in ETD is an uneven distribution of class samples (class imbalance problem) where one class has more samples as compared to the second class. When ML or deep learning models are applied on a dataset which has uneven distribution of class samples, then models become biased and ignore the minority class samples. This situation badly affects the performance of ML models. In literature, the authors mostly utilize ROS and RUS sampling techniques to solve the class imbalance problem. However, these techniques create underfitting and overfitting problems. In this article, we propose TLGAN to handle class imbalance ratio because it is specially designed for time series datasets by utilizing GRU layers.

Its objective function is based on the least-square method that computes the difference between real and fake samples and generates new samples, which have high closeness to real samples. The collected electricity theft data belongs to the time series domain. So, GRU layers are exploited to design the TLGAN model. Using the least square function, the model learns a small amount of both real theft data distribution and generated fake samples. Finally, the generated samples are concatenated with real samples and class imbalance problem is solved. The overall working mechanism of TLGAN is explained below.
We select the existing theft data as training data. The theft samples are presented as $P_{data}(x)$. A random noise or latent variable $z$ is drawn from Gaussian distribution $P_{g}(z)$. A mapping relationship is established between $P_{g}(z)$ and $P_{data}(x)$ through the GAN model. The GAN model contains two deep learning models: generator ($G$) and discriminator ($D$). The former is responsible to learn regularities from $P_{data}(x)$ distribution and generate fake samples. It takes a random variable $z$ as input from $P_{g}(z)$ and produces $G(z)$ as output. Its main goal is to fit $P_{g}(z)$ onto $P_{data}(x)$ to generate highly resembling fake samples with real theft samples and confuse the $D$ as many times as possible. The $D$ is responsible to discriminate whether input data is real or fake. It takes real theft samples and synthetic samples generated by G as input and produces output either 0 or 1, which indicates that the generated samples are either real or fake. The mathematical form of min-max equation of GAN network is given below [29].

$$\min_{G} \max_{D} V_{GAN}(D, G) = \mathbb{E}_{x \sim p_{data}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_{z}(z)} [\log (1 - D(G(z)))].$$  \hspace{1cm} (5)

where, $V_{GAN}(D, G)$ is the loss function of GAN, $\mathbb{E}_{x \sim p_{data}(x)}$ is the expected value of theft distribution and $\mathbb{E}_{z \sim p_{z}(z)}$ is the expected value of latent distribution.

**Algorithm 2** Training of TLSGAN

Data: $X_{normalized}$
1 **Variables:** Separate theft & benign samples from $X_{normalized}$. Theft: $T = \{x_{i,j}, x_{i,j+1}, x_{i,j+2}, \ldots, x_{p,n}\}$, Normal: $N = \{y_{i,j}, y_{i,j+1}, y_{i,j+2}, \ldots, y_{p,n}\}$
2 **while** Stopping condition is not met **do**
3 $t_{i} \Rightarrow$ Sample from theft distribution
4 $s_{j} \Rightarrow$ Sample from Gaussian distribution
5 $\frac{1}{n} \sum_{i=1}^{n} [\frac{1}{2} \mathbb{E}_{x \sim p_{data}(x)}[(D(x_{i}) - b)^{2}] + \frac{1}{2} \mathbb{E}_{z \sim p_{z}(z)}[(D(z_{i}) - c)^{2}]]$
6 Fix discriminator weights
7 $Z_{d} \Rightarrow$ Sample from latent space
8 $\frac{1}{n} \sum_{i=1}^{n} [\frac{1}{2} \mathbb{E}_{z \sim p_{z}(z)}[(D(z_{i}) - c)^{2}]]$
9 **end**
10 $a$ and $b$ are labels of theft and fake patterns
11 $c$ is distance that G wants to deceive $D$
12 After training of G, fake theft patterns are generated
13 $FakeSamples = G(z)$
14 $X_{BalData} = Concatenate(FakeSamples, N, T)$
15 **Result:** Return balanced dataset: $X_{BalData}$

The standard GAN network is suitable for unsupervised learning problems. It uses the binary cross-entropy function to draw a decision boundary between real and fake samples. The limitation of binary cross-entropy is that it tells whether the generated sample is real or fake but does not tell how much generated samples are far away from the decision boundary. It creates a vanishing gradient problem and stops the training process of the GAN model. In [29], the authors propose a least square generative adversarial network (LSGAN) architecture, which is an extension of the standard GAN model. It uses the least square loss instead of binary cross-entropy loss function. The LSGAN provides two benefits. The standard GAN only updates those samples, which are at wrong side of the decision boundary. The LSGAN penalizes all the samples, which are away from the decision boundary, even if the samples reside at the correct side of the boundary. During the penalization process, the parameters of $D$ and decision boundary are fixed. Now, $G$ generates samples that are closer to the decision boundary. Secondly, penalizing the samples near the decision boundary produces more changes in gradients, which solves the vanishing gradient problem. The min-max objective function of LSGAN is given in Equation (6), [29].

$$\max_{D} V_{LSGAN}(D) = \frac{1}{2} \mathbb{E}_{x \sim p_{data}(x)}[(D(x) - b)^{2}] + \frac{1}{2} \mathbb{E}_{z \sim p_{z}(z)}[(D(z) - c)^{2}],$$

$$\min_{G} V_{LSGAN}(G) = \frac{1}{2} \mathbb{E}_{z \sim p_{z}(z)}[(D(G(z)) - c)^{2}],$$

where, $V_{LSGAN}(G)$ is the loss function of LSGAN. The $a$ and $b$ are labels of real (theft data) and fake samples. $c$ is the value of distance between both samples. The $G$ needs to minimize this value in order to deceive $D$. The LSGAN is designed for generating fake images using convolutional layers. We change the internal architecture and use GRU layers instead of convolutional layers because we are working on a problem that belongs to sequential data. The training process of TLSGAN is presented in algorithm 2. We pass $X_{normalized}$ data to algorithm 2 that is obtained from algorithm 1. In the first step, variables are initialized. In steps 2 to 9, TLSGAN is trained on theft samples to generate fake theft patterns. In steps 10 to 14, the data is generated from latent distribution and passed to $G$ to produce fake theft samples. At the end, we concatenate fake samples generated by $G$, original theft samples, and normal samples and return a balanced dataset $X_{BalData}$.

**B. ARCHITECTURE OF HYBRID MODEL**

Time series data of EC has complex structure with high random fluctuations because it is affected by various factors like high load, weather conditions, big party in a house, etc. Traditional models like SVM, MLP, etc., are not ideal to learn complex patterns. The models have low DR and high FPR due to curse of dimensionality issue. In literature, different deep learning models are used to learn complex patterns from time series data.

In this article, a hybrid model is proposed, which is a combination of GoogLeNet and GRU. In [30], [31], the authors prove that hybrid deep learning models perform better than individual learners. The proposed model takes advantages of both GoogLeNet and GRU by extracting and remembering periodic features of EC dataset. The architecture of the proposed model consists of three modules: GRU, GoogLeNet and hybrid. We pass 1D data to the GRU module. Whereas, 2D weekly EC data is passed to the GoogLeNet module.
The hybrid module takes outputs of both modules, concatenates them and gives final results about having anomaly in EC patterns. The hybrid deep learning models are very efficient because they allow joint training of both models. Figure 2 shows overall structure of the proposed model. In the proposed system model, steps 1, 2 and 3 show data preprocessing phase where we handle missing values, outliers and normalize the dataset, respectively. In step 4, the class imbalance problem is solved. In steps 5 and 6, prominent features are extracted from 1D and 2D EC datasets using GoogLeNet and GRU models, respectively. Finally, in step 7, the features are extracted using the aforementioned models. The features are then merged and passed to a neural network for differentiating between normal and theft samples.

C. GATED RECURRENT UNIT

We observe that there are a lot of fluctuations in theft EC patterns as compared to normal consumers. So, 1D data is fed into GRU model to capture co-occurring dependencies in time series data. GRU is proposed by Chung et al. in 2014 to capture related dependencies in time series data. It has memory modules to remember important periodic patterns, which help to handle sudden changes in EC patterns due to non-anomalous factors like changing in weather conditions, big party in a house, weekends, etc. Moreover, it is introduced to solve the vanishing gradient problem of recurrent neural network (RNN). GRU and LSTM are considered as variants of RNN. In [32], the authors compare the performance of GRU and LSTM with RNN model on different sequential datasets. Both models outperform the RNN and solve its vanishing gradient problem. In [24], the authors from Google conduct extensive experiments on 10,000 LSTM and RNN architectures. Their final experimental results show that no single model is found that performs better than GRU. Based on the above analysis, we opt GRU to extract optimal features from EC dataset because it gives good results on sequential datasets. It has reset and update gates that control the flow of information inside the network. The update gate decides how much previous information should be preserved for future decisions. Whereas, the reset gate decides that how much past information should be kept or discarded. Equations of update and reset gates are similar to each other. However, the difference comes from weights and gates’ usage. The mathematical equations of GRU model are given below [8].

\[
\begin{align*}
    z_t &= \sigma(W_z, [h_{t-1}, x_t]), \\
    r_t &= \sigma(W_r, [h_{t-1}, x_t]), \\
    \hat{h}_t &= \tanh(W, [r_t * h_{t-1}, x_t]), \\
    h_t &= (1 - z_t) * h_{t-1} + z_t * \hat{h}_t,
\end{align*}
\]

where, \(t\), \(z_t\), \(r_t\), \(W_z\) and \(x_t\) represent time step, update gate, sigmoid function, update gate weight and current input, respectively. \(h_{t-1}\), \(\hat{h}\) and \(r_t\) are previous hidden state, candidate value, reset gate, respectively. \(W_r\) is reset gate weight, \(W\) is weight of candidate value and \(h_t\) is hidden state. The last hidden layer of GRU is presented as \(\text{Dense}_{\text{GRU}}\).

D. GoogLeNet

It is difficult to capture long-term periodicity from 1D EC data. However, periodicity can be captured if data is aligned according to weeks as explained in Section IV-E. The GoogLeNet is a deep learning model that is proposed by researchers at Google in 2014. It is designed to increase the accuracy and computational efficiency of the existing models. Its architecture is similar to the existing CNN models like LeNet-5 and AlexNet, etc. However, the core of the model is auxiliary classifiers and inception modules. Each inception module contains \(1 \times 1, 3 \times 3, 5 \times 5\) and \(7 \times 7\) convolutional filters that extract hidden or latent features from EC data. After each inception module, the output of convolutional and max pooling layers are concatenated and passed to next inception module. The auxiliary classifiers calculate training loss after 4th and 7th inception modules and add it to the GoogLeNet network to prevent it from vanishing gradient problem.

In [7], [31], the authors exploit 2D-CNN model to the extract abstract features from time series dataset. Motivated from these articles, the GoogLeNet is applied to extract latent features from EC data. The latent features increase model’s generalization ability. The 1D EC data is transformed into 2D according to weeks and is fed as input to GoogLeNet model, which has inception modules. Each inception module has max pooling and multiple convolutional layers with different filter sizes. In [7], the authors use simple CNN model to extract local patterns from EC data. In simple CNN model, multiple convolve windows of the same size move over EC patterns and extract optimal features. However, the same size of convolve windows have low ability to extract optimal features. The GoogLeNet overcomes this problem through inception modules. Different number of convolve and max pooling layers extract optimal features from EC data. Moreover, GoogLeNet has less time and memory complexity as compared to the existing deep learning models. However, it is designed for computer vision tasks that is why it has multiple inception modules to extract edges and interest points from images. For our problem, we change the architecture and use only one inception module that extracts periodicity and non-periodicity from weekly EC patterns. Finally, we use flatten and fully connected layers to attain principal features that are extracted through convolutional and max pooling layers. The last hidden layer of GoogLeNet is presented as \(\text{Dense}_{\text{GoogLeNet}}\).

E. HYBRID MODULE

GRU memorizes the periodic patterns from 1D data. Whereas, GoogLeNet captures patterns from 2D data. We combine the \(\text{Dense}_{\text{GoogLeNet}}\) and \(\text{Dense}_{\text{GRU}}\) to aggregate latent and temporal patterns. The outcome of the model is calculated through sigmoid activation function and training loss is measured using binary cross entropy.

\[
\begin{align*}
    h_{HG^2} &= (W_{HG^2} \cdot [\text{Dense}_{\text{GoogLeNet}}, \text{Dense}_{\text{GRU}}], b_{HG^2}), \\
    Y_{NTL} &= \sigma(h_{HG^2}).
\end{align*}
\]
where, $h_{HG^2}$: hidden layer of hybrid module, $W_{HG^2}$: weight of hybrid layer, $b_{HG^2}$: bias of hybrid layer, $Y_{NTL}$: output and $\sigma$: sigmoid function. We pass $X_{BalData}$ to algorithm 3 that is taken from algorithm 2. On lines 1 to 3, variables are initialized. The 1D EC data is transformed into 2D format from lines 4 to 6. On lines 7 to 17, we pass 1D data to GRU to extract time-related patterns. Whereas, 2D data is fed into GoogLeNet to retrieve periodicity and non periodicity from weekly EC patterns. On lines 18 and 19, we concatenate features of GRU and GoogLeNet and apply sigmoid activation function, which classifies theft and normal EC patterns.

**F. PERFORMANCE METRICS**

One of the main challenges of ETD is a class imbalance problem where classifiers become biased towards the majority class and ignore the minority class. Therefore, the selection of suitable measures is necessary to evaluate the performance of
Algorithm 3 Training of HG²

Data: EC dataset: X_{BalData}
1 Data in 1D format
2 \(X_{ID} = \{x_{ij}, x_{i,j+1}, x_{i,j+2}, \ldots, x_{m,n}\}\)
3 \(m = 42372, n = 1034\)
4 Convert data in 2D format
5 \(Z = \begin{bmatrix} x_{1,1} & \cdots & x_{1,k} \\ \vdots & \ddots & \vdots \\ x_{j,1} & \cdots & x_{j,k} \end{bmatrix}\)
6 \(j = 147, k = 7\)
7 Pass \(X_{ID}\) to GRU
8 \(z_t = \sigma(W_z, [h_{t-1}, x_t])\)
9 \(r_t = \sigma(W_r, [h_{t-1}, x_t])\)
10 \(h_t = \tanh(W, [r_t \cdot h_{t-1}, x_t])\)
11 \(h_t = (1 - z_t) \cdot h_{t-1} + z_t \cdot \hat{h}_t\)
12 \(\text{Dense}_{\text{GRU}} = \text{relu}(W \cdot h_t, b)\)
13 Pass \(Z\) to GoogLeNet
14 \(Z[a, c] = (Z[a, c] = \sum_{k} f[j, k]Z[a - j, c - k]\)
15 \(a, c \Rightarrow \text{dimension of output matrix}\)
16 \(F_{\text{GoogLeNet}} = \text{flatten}(Z)\)
17 \(\text{Dense}_{\text{GoogLeNet}} = [W \cdot F_{\text{GoogLeNet}} + b]\)
18 \(h_{\text{HG}^2} = (W_{\text{HG}^2} \cdot [\text{Dense}_{\text{GRU}}, \text{Dense}_{\text{GoogLeNet}}] + b)\)
19 \(b \Rightarrow \text{bias term}\)
20 \(Y_{\text{NTL}} = \sigma(h_{\text{HG}^2})\)

Result: \(Y_{\text{NTL}}\)

classifiers for both classes. We opt ROC-AUC and PR-AUC as performance metrics. The ROC-AUC is retrieved by plotting true positive rate (TPR), also known as recall, on y-axis and FPR on x-axis. It is a convenient diagnostic tool because it is not biased towards minority and majority classes. Its value lies between 0 and 1. Although, ROC-AUC is a good performance measure, however, it does not consider precision of a classifier and does not give equal importance to both classes. Additionally, test dataset has imbalance nature, so we decided to take into account PR-AUC for performance evaluation of the classifiers [8]. PR-AUC is a ratio between recall and precision. The precision measures the percentage of correctly identified number of electricity thieves. The maximization of precision increases recovery revenue of utility. The recall calculates percentage of electricity thieves on suspicious list. High scores of precision and recall are very important for accomplishing the goals of a utility.

VI. EXPERIMENTS AND RESULTS ANALYSIS

In this paper, all models are trained and tested on SGCC dataset. The description of the dataset is given in Section IV-A. We use Google Colab to train deep learning and ML models by taking advantage of distributed clustering computing. Deep learning models are implemented through TensorFlow, which is a deep learning library. Moreover, conventional models are fitted through the scikit-learn library.
Table 4 presents classification accuracy and execution time of different data generation techniques. We compare the performance of proposed TLSGAN with current variants of SMOTE: SVM_SMOTE, Borderline SMOTE, SMOTE_TomekLinks, SMOTE_ENN and ADASYN. TLSGAN generates new theft samples that increase the classification accuracy of the proposed model. As explained above, the generated samples have a high resemblance with real theft samples that reduces the overfitting problem, which occurs in other oversampling techniques and increases the model generalization and robustness properties. The execution time of TLSGAN is more than ROS, RUS, Borderline-SMOTE and SMOTE. While it is less than SVM_SMOTE, SMOTE_TomekLinks, SMOTE_ENN and ADASYN. The running time of TLSGAN depends upon the number of hidden layers and the sampling rate of a dataset. The execution time of SVM_SMOTE, Borderline-SMOTE, SMOTE_TomekLinks, SMOTE_ENN, ADASYN and SMOTE depends upon the number of samples and features in a dataset. Whereas, the execution time of RUS and ROS does not significantly change with large datasets because they simply select samples from the dataset and duplicate or remove them. SMOTE_TomekLinks and SMOTE_ENN techniques take too much time because they perform under sampling and oversampling steps to remove redundant samples from the datasets.

| Techniques          | Execution time (s) | Accuracy (%) |
|---------------------|--------------------|--------------|
| TLSGAN              | 61.61              | 95           |
| SVM_SMOTE           | 177                | 88           |
| Borderline_SMOTE    | 96                 | 90           |
| SMOTE_TomekLinks    | 71.61              | 93           |
| SMOTE_ENN           | 95.78              | 89           |
| ADASYN              | 71.61              | 93           |
| SMOTE               | 9.18               | 81           |
| ROS                 | 0.12               | 88           |
| RUS                 | 0.5                | 89           |

Table 5 presents the hyperparameters setting of GRU model. The GRU model has update and reset gates to regulate the flow of information throughout the network. These gates prevent the model from vanishing gradient problem and reduce its chances of sticking in local minima problem. Moreover, these gates increase the model’s overall performance by extracting the optimal temporal features from the EC dataset, which have time-related dependencies after certain intervals. Table 5 presents the hyperparameters setting of GRU model.

C. PERFORMANCE ANALYSIS OF GoogLeNet

Figures 5a, 5b and 5c show the performance of the GoogLeNet model. The Figure 5a shows PR curve of GoogLeNet model. PR curve provides a good analysis of the model’s performance because it gives equal weights to both normal and abnormal samples. The model obtains good PR curves on training and testing datasets, which indicate that it learns patterns of both normal and abnormal samples appropriately during the training phase. Figure 5b shows the model’s performance using the ROC curve and ROC-AUC performance indicators. These indicators evaluate how much a model is good in predicting the positive class. GoogLeNet achieves 95.7% and 94.2% AUC values on training and testing datasets, respectively, that are more than AUC values of the GRU model. Moreover, loss and accuracy of the model on training and testing datasets can be seen in Figure 5c. We visualize the model’s performance with more than 20 epochs. However, we observe that there are more number of fluctuation in training and testing curves of
accuracy and loss, which indicate the model’s instability on more than 20 epochs. Due to the above mentioned reasons, model is trained only on 20 epochs that give good results and save our computational resources. In this model, data shape is transformed according to weeks to learn periodic patterns and extract optimal features through convolution and max pooling layers. The max pooling layers reduce data dimensionality that increases the model’s convergence speed. Moreover, dropout layers are used to reduce overfitting problem and increase generalization property. Table 6 presents the hyperparameters setting of GoogLeNet.

### D. PERFORMANCE ANALYSIS OF HYBRID HG\(^2\) MODEL

In this Section, the performance of HG\(^2\) model is compared with stand-alone deep learning models. Figures 6a, 6b and 6c show the HG\(^2\) model’s performance using different performance measures. HG\(^2\) achieves 97.8% and 95.7% ROC-AUC values on training and testing datasets, respectively, that are more than GRU and GoogLeNet models. Figure 6c shows loss and accuracy curves on training and testing datasets that are better than curves of GRU and GoogLeNet models, which are presented in Figures 4c and 5c. In [30] and [31], the authors prove that a hybrid deep learning model performs better than individual learners and achieves better convergence speed, takes less computational time and extracts optimal features. The GRU layers extract time related patterns through update and reset gates. Whereas, GoogLeNet model has inception module, which contains max pooling and multiple convolution layers with different filter sizes. These layers reduce computational complexity and extract latent and abstract patterns using local receptive fields and weight sharing mechanism. The Keras library is used to concatenate extracted optimal features of both GoogLeNet and GRU classifiers. Finally, these concatenated features have properties of both individual learners that provide better learning to HG\(^2\) model. Although, it gives low performance. However, when we combine it with GoogLeNet model, then the overall performance is improved. The combined model has the ability to learn better patterns from EC data. The proposed hybrid model ignores the weak points of both GRU and GoogLeNet and uses the strong points of both. This is the reason why the poor performance of GRU does not affect the overall performance of HG\(^2\). Table 7 shows its hyperparameters setting.

### E. COMPARISON WITH BENCHMARK CLASSIFIERS

In this Section, the performance of HG\(^2\) is compared with existing state-of-the-art deep learning and ML classifiers.

1) **WIDE AND DEEP CONVOLUTIONAL NEURAL NETWORK (WDCNN)**

It is proposed in [7] to identify normal and abnormal patterns from EC data. The wide component is equivalent to MLP...
module that is used to extract global knowledge from data. Whereas, CNN is leveraged to attain periodic patterns from weekly EC data. We use same dataset and hyperparameters setting to compare the model with our proposed model.

2) HYBRID MULTILAYER PERCEPTRON AND LONG SHORT TERM MEMORY MODEL
In [8], the authors propose a hybrid model where they integrate the benefits of both LSTM and MLP. They pass EC data to LSTM to extract periodic patterns. Whereas, smart meter data is fed to the MLP model to retrieve non-sequential information. They concatenate both models through Keras library and prove that a hybrid model is better than a single model. We use the same number of hyperparameters and dataset settings as utilized in [8] to build a hybrid LSTM-MLP model.

3) NAIVE BAYES CLASSIFIER (NB)
It is a statistical classification technique that is based on bayes theorem. It assumes that there is no relationship between input features and predicts the unknown class using a probability distribution. It has high accuracy and speed on large datasets. Moreover, it has many applications in the real world: spam filtering, sentiment analysis, text classification, recommendation systems, etc. The NB has different versions according to the nature of a dataset. We utilize Gaussian NB to classify normal and abnormal data points in the EC dataset because it is specially designed for the prediction of continuous values.

4) SUPPORT VECTOR MACHINE
It is a well-known classifier in ETD. It can classify both linear and non-linear data. It exploits radial, sigmoid, Gaussian, etc., kernels to transform non-linear data into a linear format and then draw a decision boundary between electricity thieves and normal consumers. However, its computational time is high for large datasets. In [2], the authors use SVM to classify benign and theft consumers. We use radial basis function kernel (RBF) due to the non-linearity of data and different values of C parameter. After several iteration, 100 is found to be the optimal value of C where SVM gives good results.

5) LOGISTIC REGRESSION (LR)
It is a supervised ML algorithm used for the binary classification task. It is just like one layered neural network. For probability of having NTL, it multiplies the input features and predicts the unknown class using a sigmoid function to generate output between 0 and 1. It has different solver methods: newton’s method, stochastic average gradient and sparse stochastic average gradient (SGA). However, Newton’s method gives best results that are mentioned in Table 8.

Results: We compare the performance of the proposed HG model with different state-of-the-art classifiers. The same training and testing datasets are used for LR, NB, MLP and SVM. We use RBF kernel for SVM due to the non-linearity of data. Moreover, number of samples and dimensionality of data is reduced because SVM requires high computational time for large datasets. In [8], the authors use sequential and non-sequential data for LSTM and MLP, respectively. However, we do not have availability of non-sequential information that is why only sequential information is fed into MLP and LSTM models. The hybrid of both models gives good results and achieves 95% and 94% ROC-AUC and PR-AUC, respectively.

In [7], the sequential data is passed as input to MLP model for retrieving global patterns from EC data. Whereas, 2D stacked data is given to CNN model to extract periodic patterns from weekly EC data. The WDCNN achieves 92% and 88% ROC-AUC and PR-AUC, respectively, which are more than as compared to ROC-AUC and PR-AUC of conventional ML models.

The proposed HG model outperforms hybrid and other ML models because it extracts periodic and abstract patterns from EC data using GRU and convolutional layers. As discussed earlier, the GRU layers have update and reset gates that learn important patterns and remove redundant values. These gates control the flow of information and improve the overall performance of proposed model. GoogLeNet has an inception module that contains max pooling and multiple convolutional layers with different filter sizes. These layers extract those patterns that cannot be retrieved through human knowledge. These abstract or latent patterns are combined with features that are extracted by GRU model. Due to a combination of optimal features, HG attains 96% and 97% ROC-AUC and PR-AUC values that are more than all above explained classifiers.

Table 8 shows comparison results of proposed model and all other classifiers on different training ratios of datasets. The deep learning models are sensitive to the size of training data. The performance of these models increases with the growing amount of training data. However, this is not true for conventional ML models and their performance increases according to the power law. After a certain point of training on data, their performance does not improve [33]. However, HG maintains superiority on other deep learning models and gives better performance on different training ratios on SGCC dataset. Both SVM and NB give good results on balanced and large datasets. However, in our case, these models perform poorly due to the following reasons. The SVM does not perform well on noisy data and NB’s performance is affected by continuous values because it assumes an independent relationship between features. For MLP-LSTM, WDCNN and HG, if performance is not increasing or decreasing then it means that we must perform hyperparameter tuning on training data to improve results.

F. MAPPING AMONG LIMITATIONS, SOLUTIONS AND VALIDATIONS
Table 9 shows mapping of limitations, solutions and their validations. L1 describes about class imbalance problem where classifiers are biased towards majority and ignores
TABLE 8. Comparison of HG\(^2\) with existing techniques.

| Dataset | Training data = 80% | Training data = 60% | Training data = 50% |
|---------|--------------------|--------------------|--------------------|
| Methods | ROC-AUC (%) | PR-AUC (%) | ROC-AUC (%) | PR-AUC (%) | ROC-AUC (%) | PR-AUC (%) |
| SVM     | 77            | 64            | 77            | 64            | 78            | 64            |
| LR      | 88            | 82            | 87            | 80            | 90            | 83            |
| NB      | 50            | 52            | 50            | 54            | 51            | 59            |
| MLP     | 88            | 82            | 87            | 79            | 86            | 77            |
| MLP-LSTM| 95            | 94            | 92            | 90            | 88            | 82            |
| WDCNN   | 92            | 88            | 91            | 89            | 95            | 93            |
| HG\(^2\)| 96            | 97            | 93            | 91            | 88            | 85            |

TABLE 9. Mapping table.

| Limitations                               | Solutions     | Validations                                      |
|-------------------------------------------|---------------|-------------------------------------------------|
| L1: Class imbalance problem               | S1: TLSGAN    | V1: The proposed model achieves 96% PR-AUC that indicates model is not biased toward majority class, which is shown in Figure 6a. |
| L2: RUS removes important information from data | S1: TLSGAN does not remove information from dataset | V2: Result of TLSGAN and RUS are given in Table 4. Figure 6c shows that model is not stuck into underfitting problem. |
| L3: ROS causes overfitting problem        | S1: TLSGAN reduces the overfitting problem of ROS by generating fake samples, which have high resemblance with real samples | V3: It achieves good PR curve and ROC curve on training and testing datasets as shown in 6a and 6b |
| L4: SMOTE causes overfitting problem      | S1: TLSGAN overcomes overfitting issue of SMOTE and ADASYN | V4: Table 4 shows accuracy of SMOTE, ADASYN and TLSGAN. The proposed model attains good PR curve and ROC curve on training and testing data, which indicates that model is not stuck into overfitting problem. |
| L5: Curse of dimensionality increases model complexity and reduces model generalization ability | S2: GRU and GoogLeNet are used to extract features from sequence data (1D data) | V5: In Figures 6a, 6b and 6c, proposed model achieves good results, which indicate that GoogLeNet and GRU extracts optimal temporal patterns from EC dataset. |
| L6: High FPR and overfitting issue        | S3: Dropout and batch normalization layers are used to reduce FPR and overfitting problem | V6: We evaluate the model on training and testing data, which achieve FPR score that is lower than FPR of the existing models. |

minority class that increases the FPR score. S1 solution is proposed for L1. In S1, the TLSGAN is used to handle class imbalance problem. As shown in Table 9, V1 is validation of S1. The proposed model achieves 96% PR-AUC score that indicates model is not biased toward majority class. Moreover, it achieves 4% FPR score that is acceptable for a utility. In L2, RUS randomly removes samples of majority class to balance ratio of theft and normal samples. However, it discards the useful information from data that causes overfitting problem. S2 solution is proposed to tackle L2. In S2, the TLSGAN is a deep learning technique that is designed to generate fake samples, which have resemblance with real samples. So, this technique does not remove useful information from data and solves drawbacks of RUS. V2 validates the S2. Figure 6c shows that model is not stuck in underfitting problem. In L3 and L4, the existing data sampling techniques generate duplicated copies of minority class to solve class imbalance problem. These techniques are designed for tabular data and not for time series data. So, they face overfitting issue on time series data. TLSGAN is specially designed to generate fake samples of time series datasets that have severe class imbalance problem. TLSGAN uses supervised and unsupervised loss functions and generates samples that resemble with actual data and also preserves time related patterns. The performance of TLSGAN is compared with advanced variants SMOTE techniques. V3 and V4 are validations of S1. Table 4 shows the comparison between different data sampling techniques, which shows that accuracy of TLSGAN is more than benchmark data augmentation techniques. Figure 6c indicates that HG\(^2\) attains good loss and accuracy curves on training and testing datasets. Moreover, proposed model achieves good PR curve that can be seen in Figure 6a. L5 are issues that occur due to curse of dimensionality. The GoogLeNet is used to capture weekly periodicity from 2D data. Whereas, the GRU is leveraged to capture long term and short term features from 1D data. In S2, GRU and GoogLeNet extract temporal and latent patterns and pass them to a hybrid neural network to classify theft and normal samples. V5 is validation of S2. Figures 6a, 6b and 6c show performance of proposed model through accuracy, loss, PR and ROC curves, which indicate that GoogLeNet and GRU extracts optimal temporal patterns from EC dataset and transfer them to hybrid module. Due to these optimal features, HG\(^2\) achieves 96% and 97% ROC-AUC and PR-AUC scores, respectively that are more than existing techniques, which are mentioned in Table 8. L6 is about high FPR and overfitting problem. We know that utilities cannot bear high FPR due to limited budget for on site inspection. In S4, dropout and batch normalization layers are leveraged to solve overfitting problem and reduce the FPR score. V6 validates S4 by computing FPR. The proposed model achieves 4% FPR that is lower than as compared to FPR of all existing models. VII. CONCLUSION

In this article, we propose a model to detect NTLs in the electricity distribution system. The proposed model is a hybrid of GRU and GoogLeNet. The GRU is used to extract temporal patterns from time series dataset. Whereas, the GoogLeNet is exploited to attain latent patterns from the weekly stacked
EC dataset. The performance of proposed model is evaluated on realistic EC dataset that is provided by SGCC, the largest smart grid company in China. The simulation results show that HG^2 outperforms the benchmark classifiers: WDCNN, MLP-LSTM, MLP, LR, NB and SVM. Moreover, the class imbalance problem is a severe issue in ETD. The TSLGAN is proposed that consist of GRU and dense layers to tackle the class imbalance problem. The TSLGAN generates fake samples, which have high resemblance with real world theft samples. The model is evaluated using suitable performance measures: ROC-AUC and PR-curve. The results of these measures indicate that HG^2 gives better performance as compared to ML and deep learning models by achieving 96% and 97% ROC-AUC and PR-AUC values, respectively.

In near future, we plan to implement the proposed model as an NTLs detector in an electricity distribution company in Pakistan to classify normal and theft samples.

**DATASET AVAILABILITY**

Dataset used in this study is publically available at: https://github.com/henryRDLab/ElectricityTheftDetection.

**REFERENCES**

[1] L. G. Arango, E. Deccache, B. D. Bonato, H. Arango, P. F. Ribeiro, and P. M. Silveira, “Impact of electricity theft on power quality,” in Proc. 17th Int. Conf. Harmon. Qual. Power (ICHQP), Oct. 2016, pp. 557–562.

[2] P. Jokar, N. Arianpour, 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.

[3] R. Punmiya and S. Cho, “Energy theft detection using gradient boosting theft detector with feature engineering-based preprocessing,” IEEE Trans. Smart Grid, vol. 10, no. 2, pp. 2326–2329, Mar. 2019.

[4] C.-H. Lo and N. Ansari, “CONSUMER: A novel hybrid intrusion detection system for distribution networks in smart grid,” IEEE Trans. Emerg. Topics Comput., vol. 1, no. 1, pp. 33–44, Jun. 2013.

[5] B. Khoob and Y. Cheng, “Using RFID for anti-theft in a Chinese electrical supply company: A cost-benefit analysis,” in Proc. Wireless Telecommun. Symp. (WTS), Apr. 2011, pp. 1–6.

[6] S. Amin, G. A. Schwartz, and H. Tembine, “Incentives and security in electricity distribution networks,” in Proc. Int. Conf. Decis. Game Theory Comput. Security, Berlin, Germany: Springer, 2012, pp. 264–280.

[7] 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.

[8] M.-M. Buzau, J. Tejedor-Aguilera, P. Cruz-Romero, and A. Gomez-Exposito, “Hybrid deep neural networks for detection of non-technical losses in electricity smart meters,” IEEE Trans. Power Syst., vol. 35, no. 2, pp. 1254–1263, Mar. 2020.

[9] M. M. Buzau, J. Tejedor-Aguilera, P. Cruz-Romero, and A. Gomez-Exposito, “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.

[10] M. N. Hasan, R. N. Toma, A.-A. Nahid, M. M. M. Islam, and J.-M. Kim, “Electricity theft detection in smart grid systems: A CNN-LSTM based approach,” Energies, vol. 12, no. 17, p. 3310, Aug. 2019.

[11] 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.
FAISAL SHEHZAD received the B.S.(S.E.) degree in software engineering from Government College University Faisalabad (GCUF), Faisalabad, Pakistan, in 2018. He is currently pursuing the M.S. degree in computer science with the Communication over Sensors (ComSens) Research Laboratory, Department of Computer Science, COMSATS University Islamabad, Islamabad, Pakistan, under the supervision of Dr. Nadeem Javaid. He has five research publications in well-reputed international journals and conferences. His research interests include data science, smart grid, blockchain, and financial market.

NADEEM JAVAID (Senior Member, IEEE) received the bachelor’s degree in computer science from Gomal University, Dera Ismail Khan, KPK, Pakistan, in 1995, the master’s degree in electronics from Quaid-i-Azam University, Islamabad, Pakistan, in 1999, and the Ph.D. degree in computer science from the University of Paris-Est, France, in 2010. He is currently an Associate Professor and the Founding Director of the Communications over Sensors (ComSens) Research Laboratory, Department of Computer Science, COMSATS University Islamabad, Islamabad. He has supervised 137 master’s theses and 24 Ph.D. theses. He has authored over 900 articles in technical journals and international conferences. His research interests include energy optimization in smart grids and wireless sensor networks using data analytics and blockchain. He was a recipient of the Best University Teacher Award from the Higher Education Commission of Pakistan in 2016 and the Research Productivity Award from Pakistan Council for Science and Technology in 2017. He is also an Associate Editor of IEEE Access and an Editor of the Sustainable Cities and Society.

AHMAD ALMOGREN (Senior Member, IEEE) received the Ph.D. degree in computer science from Southern Methodist University, Dallas, TX, USA, in 2002. He is currently a Professor with the Computer Science Department, College of Computer and Information Sciences (CCIS), King Saud University (KSU), Riyadh, Saudi Arabia, where he is also the Director of the Cyber Security Chair, CCIS. Previously, he worked as the Vice Dean of the development and quality with CCIS. He also served as the Dean for the College of Computer and Information Sciences and the Head for the Academic Accreditation Council, Al Yamamah University. His research interests include mobile-pervasive computing and cyber security. He served as the General Chair for the IEEE Smart World Symposium and a Technical Program Committee Member for numerous international conferences/workshops, such as IEEE CCNC, ACM BodyNets, and IEEE HPCC.

ABRAR AHMED was born in Pakistan, in 1985. He received the B.S. degree in computer engineering from the COMSATS Institute of Information Technology, Abbottabad, Pakistan, in 2006, the M.S. degree from Lancaster University, U.K., in 2008, and the Ph.D. degree in electrical engineering from the COMSATS Institute of Information Technology, Islamabad, in 2017. Since 2006, he has been associated with the COMSATS Institute of Information Technology, Islamabad, where he currently holds the position of an Assistant Professor. His research interests include wireless channel modeling and characterizing, smart antenna systems, nonorthogonal multiple access techniques, and adaptive signal processing.

SARDAR MUHAMMAD GULFAM received the M.S. degree in computer engineering from Tampere University of Technology, Finland, in 2010, and the Ph.D. degree in electrical engineering from the COMSATS Institute of Information Technology, Islamabad, in 2017. He is currently working as a Researcher in wireless communication.

AYMAN RADWAN (Senior Member, IEEE) received the Ph.D. degree from Queen’s University, Kingston, ON, Canada, in 2009. He is currently a Senior Research Engineer (Investigador Auxiliar) with the Instituto de Telecomunicações, University of Aveiro, Aveiro, Portugal. He is mainly specialized in coordination and management of EU funded projects. He participated in the coordination of multiple EU projects. He is also the Project Coordinator of the CELTIC+ Project MUSCLES and participating in the coordination of ITN-SECRET. He has also been the Technical Manager of the FP7-C2POWER Project and the Coordinator of the CELTIC+ Green-T Project. His current research interests include the Internet of Things, 5G, and green communications.