An AI Explained Data-Driven Framework for Electricity Theft Detection with Optimized and Active Machine Learning

Nadeem Javaid[a,*,1], Muhammad Hasnain[a] and Muhammad Ammar[a]

[a]ComSens Lab, International Graduate School of Artificial Intelligence, National Yunlin University of Science and Technology, Douliu, Yunlin 64002, Taiwan

ARTICLE INFO

Keywords:
Electricity Theft Detection
Stochastic Gradient Descent
Cuckoo Search Optimization
Active Machine Learning
Explainable Artificial Intelligence
Local Interpretable Model-agnostic Explanations
Shapley Additive Explanations

ABSTRACT

Electricity theft is a major problem that causes significant financial losses and inefficient power distribution. Effective theft detection systems play a critical role in detecting fraudulent consumption patterns. However, the performance and generalization of traditional theft detection systems are hindered by issues such as class imbalance, lack of labeled data, suboptimal hyperparameter tuning, and limited model interpretability. To overcome these issues, we propose a novel framework that combines active learning and metaheuristic optimization to enhance theft detection performance. Initially, the proposed framework addresses the data imbalance in the State Grid Corporation of China dataset by employing localized randomized affine shadow sampling. Next, two models are proposed to increase classification accuracy: Active Stochastic Gradient Descent (ASGD) and Cuckoo Stochastic Gradient Descent (CSGD). The ASGD uses entropy-based active learning to prioritize informative samples, whereas CSGD incorporates cuckoo search optimization to improve parameter tuning. The proposed ASGD and CSGD models show significant improvements of 36.67% and 35%, respectively, over the baseline SGD in accuracy, demonstrating enhanced performance in electricity theft detection. The experimental results demonstrate that ASGD and CSGD outperform state-of-the-art models with an improvement score of 6.57% and 7.89% in accuracy, 7.89% and 9.21% in F1-score, and 7.14% and 8.33% in the precision-recall area under the curve. Furthermore, the results of the proposed models are validated using a 10-fold cross-validation technique to ensure their reliability. Additionally, the statistical significance of ASGD and CSGD is confirmed using a t-test. Finally, two explainable artificial intelligence methods: local interpretable model-agnostic explanations and Shapley additive explanations, are employed to uncover the interpretability and explainability of the proposed models' predictions. The proposed framework is useful for detecting electricity consumption anomalies as it enhances both classification performance and model interpretability, ensuring more reliable predictions.

1. Introduction

Electricity theft (ET) is a pervasive issue that significantly affects the reliability and sustainability of smart grids globally. It leads to massive financial losses for power companies, disrupts energy distribution, and creates operational challenges [1]. In developing nations such as India and Pakistan, where infrastructure may be deficient, the extent of ET can reach alarming levels, potentially accounting for up to 20% of the total energy generated [2]. This not only results in substantial financial implications but also causes grid instability, increased system congestion, and a decline in service quality for legitimate customers [3].

Traditional methods of detecting ET predominantly rely on hardware-based solutions such as smart meters and sensors [4, 5]. These methods, while effective to some extent, have notable limitations including high installation and maintenance costs, slow response times to evolving fraudulent tactics, and limited scalability [6]. In addition, these systems often fail to adapt to new theft techniques and are incapable of handling the increasing complexity of electricity consumption patterns. Furthermore, those structures frequently lose their ability to scale and adapt, which makes them less capable of handling the variety and sophistication of techniques used by energy thieves. However, a new concept of Advanced Metering Infrastructure (AMI) has been introduced in smart grids [7, 8]. Moreover, the data generated from these traditional systems is often unbalanced, with a disproportionate number of honest cases compared to fraudulent ones, making the detection process unreliable and inefficient. Figure 1 illustrates the comprehensive smart grid architecture, encompassing transmission lines, distribution systems, factories, smart homes, nuclear power, hydropower, offices, solar power, Electric Vehicles (EVs), and wind power, all integrated to optimize energy distribution and management.

Recent advances in Machine Learning (ML) and Artificial Intelligence (AI) offer promising solutions to the challenges posed by ET. However, the current ML models used for ET Detection (ETD) still face significant hurdles, particularly high class imbalance, limited labeled datasets, and difficulty in selecting optimal hyperparameters. Additionally, many existing models lack interpretability and explainability, which are crucial factors for their acceptance in practical applications, especially in sensitive fields like ETD. This absence of transparency in model decisions leads to a lack of trust and adoption by utilities and other stakeholders who require explainable outcomes.

In this study, we propose a novel solution to improve ETD through an Active Learning (AL) and optimization-based ML framework. Our method combines advanced AL

*Corresponding Authors: javaidn@yuntech.edu.tw, www.njavaid.com
ORCID(s):
[1] https://orcid.org/0000-0003-3777-8249

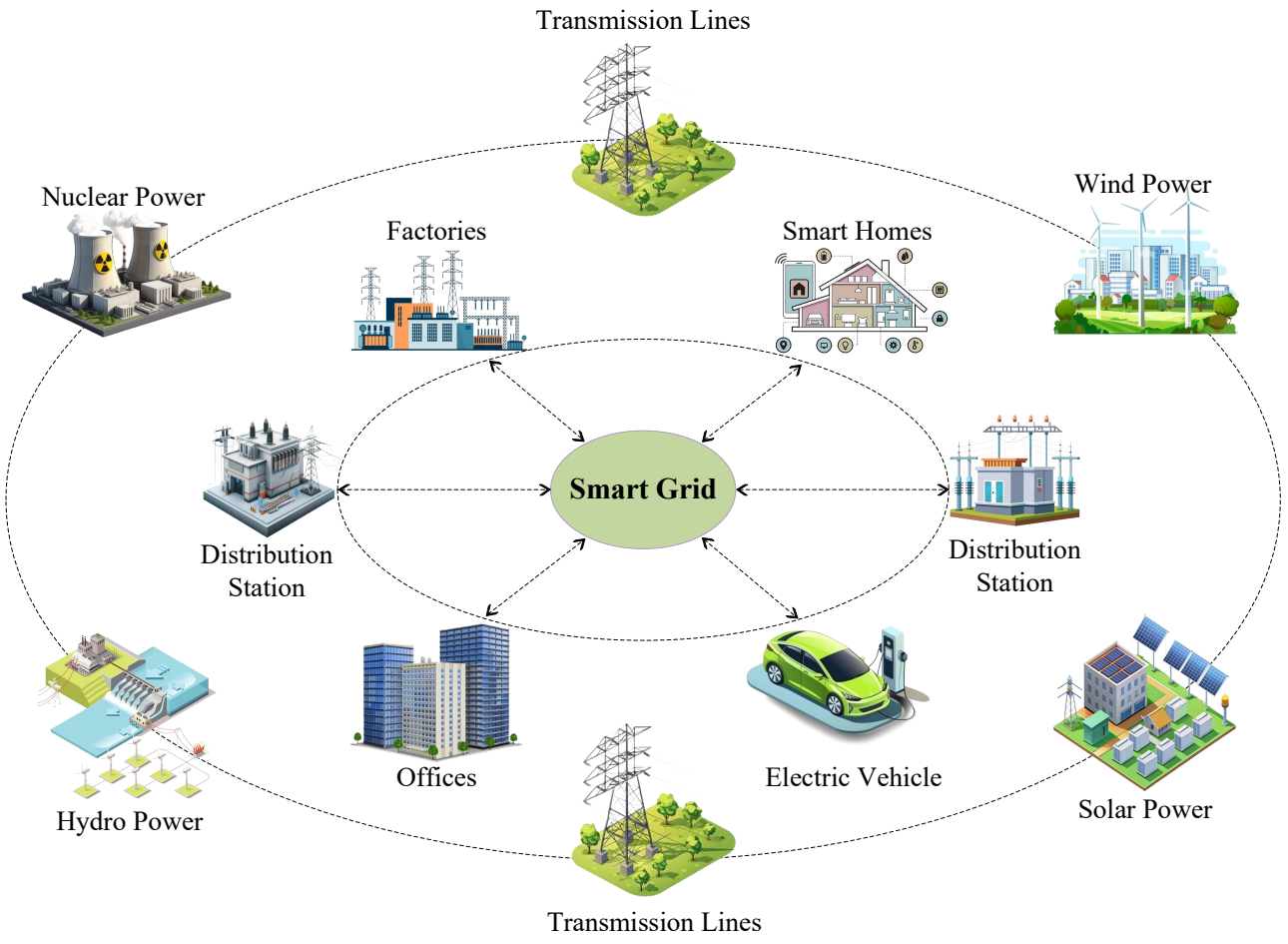


Figure 1: A generic topology of a Smart grid for electric supply

and optimization with Explainable AI (XAI) techniques to enhance both the accuracy and efficiency of ETD systems. The primary goal of this approach is to address the shortcomings of traditional systems by providing scalable, cost-effective, and interpretable solutions. We demonstrate how the proposed framework outperforms existing approaches, offering improved performance metrics and greater adaptability to real-world challenges, thus contributing to the effective management of ET in modern smart grids. A list of abbreviations and symbols is provided in Table 1.

1.1. Research Gaps and Contributions

The application of ML classifiers for ETD has made significant advancements in the smart grid domain. However, there are still several research gaps, including the high-class imbalance, poor learning performance, lack of interpretability and explainability, insufficient model generalization, and statistical significance with existing methods. To address these gaps, this paper proposes a data-driven AI-explained framework that integrates key advancements in ETD.

1.1.1. Addressing Class Imbalance in Imbalanced Dataset using Localized Randomized Affine Shadow Sampling

Conventional ML classifiers often suffer from biased learning when fraudulent cases are less common than normal ones, which leads to limited detection performance. The utilization of numerous cutting-edge data balancing techniques in existing research often results in overfitting issues or information loss, which presents significant research challenges [9, 10]. These techniques' static working makes them less applicable in real-world scenarios because they are incapable of adapting to changing ET patterns' behaviour.

In this study, the highly imbalanced State Grid Corporation of China (SGCC) dataset is oversampled using the Localized Randomized Affine Shadow sampling (LoRAS) technique. This method generates synthetic fraudulent samples within the distributional boundaries of fraudulent consumers, mimicking real ET behaviors. The goal is to mitigate the class imbalance problem, reduce bias, and preserve meaningful variation in electricity consumption. This method is detailed in Section 4.1 (Data Balancing Performance Analysis), where its effectiveness in mitigating class imbalance is analyzed and compared.

Table 1
List of abbreviations and symbols with their descriptions

Abbreviation	Description	Symbol	Symbol Description
AEs	AutoEncoders	b	Bias parameter of the model
AL	Active Learning	β	Step size scaling factor in Lévy flight
AMI	Advanced Metering Infrastructure	$f(c_i) = z_i$	Mapping function for class c_i
ASGD	Active Stochastic Gradient Descent	$\mathcal{H}(x)$	Entropy of the predicted class distribution for sample x
CS	Cuckoo Search	$L(\lambda)$	Lévy flight distribution parameterized by λ
CSGD	Cuckoo Stochastic Gradient Descent	$\mathcal{L}(y, \hat{y})$	Log-loss function measuring prediction error
ET	Electricity Theft	$\mathcal{M}(x)$	Margin between the top two predicted class probabilities.
ETD	Electricity Theft Detection	N	Total number of instances or cuckoo nests
FN	False Negative	η	Learning rate used in SGD
FP	False Positive	$\mathcal{P}(x)$	Probability distribution over all classes for sample x
K-FCV	K-Fold Cross-Validation	T	Maximum number of iterations for optimization
LIME	Local Interpretable Model-agnostic Explanations	X	Training data (features)
LoRAS	Localized Randomized Affine Shadow sampling	x_i	Current solution (nest) i in Cuckoo Search
ML	Machine Learning	x_i^{new}	Newly generated solution (nest) i using Lévy flight
NTLs	Non-Technical Losses	x_i^{new}	Newly generated solution (nest) i using Lévy flight
PR-AUC	Precision-Recall Area Under the Curve	x_{syn}	Synthetic sample generated using interpolation
ROC-AUC	Receiver Operating Characteristics Area Under the Curve	y	True labels for the training data
SG	Smart Grids	$\hat{y}$	Predicted probability for a sample
SGD	Stochastic Gradient Descent	λ	Lévy parameter for generating Lévy flight or regularization hyperparameter
SHAP	SHapley Additive exPlanations	$\sigma(z)$	Sigmoid activation function defined as $\sigma(z) = \frac{1}{1+e^{-z}}$
TN	True Negative	θ	Weight parameters of the model
TP	True Positive	$\|\theta\|_2$	L_2-norm of the weight vector θ, used for regularization

1.1.2. Improving Learning Performance for Electricity Theft Detection with Active Stochastic Gradient Descent and Cuckoo Stochastic Gradient Descent

In recent years, the majority of existing approaches have heavily depended on traditional ML classifiers that often struggle to capture complex, non-linear relationships in high-dimensional electricity consumption data [9, 11, 12, 13, 14]. These classifiers' inability to use AL methods results in ineffective ETD as training neglects the most informative cases. Additionally, the majority of existing research produces undesirable model output because it lacks efficient optimization techniques for hyperparameter mutation, leading to ineffective exploration when dealing with high-dimensional data and struggles with early convergence [15, 16].

However, in this study, two new models named Active Stochastic Gradient Descent (ASGD) and Cuckoo Stochastic Gradient Descent (CSGD) are proposed to improve ETD efficiency. The ASGD, used in conjunction with entropy-based AL, iteratively identifies the most informative instances from the dataset to enhance model training. On the other hand, the CSGD optimizes the base Stochastic Gradient Descent (SGD) by incorporating Cuckoo Search (CS), which improves both exploration and convergence, ultimately resulting in better classification accuracy and parameter optimization. This contribution is addressed in Sections 4.2 and 4.3 (Justification for Benchmark and Proposed Models and Comparative Analysis of ASGD and CSGD with Benchmark Models). These sections detail the justification for the proposed ASGD and CSGD models, focusing on their improved performance over benchmark models and their ability to handle imbalanced datasets and improve learning outcomes in ETD.

1.1.3. Ensuring Models' Generalization Across Different Subsets of Data with 10-Fold Cross-Validation

Another significant gap in current approaches is the lack of comprehensive testing to evaluate the model's generalization across different subsets of data. This limitation restricts the generalization of the results. Validation across different dataset subsets is essential to ensure the robustness and applicability of the model in real-world scenarios and to assess its stability under various conditions.

To overcome this, we evaluate the performance of the proposed models and ensure their generalizability with the 10-Fold Cross-Validation (10-FCV) technique. This guarantees that the outcomes of the proposed models are not overfitted and can be reliably applied to unseen data. This contribution is covered in Section 4.7 (10-Fold Cross-Validation of Proposed ASGD and CSGD Models). It highlights the application of the 10-FCV technique to assess the generalization and robustness of the ASGD and CSGD models, ensuring their reliability in various data subsets.

1.1.4. Statistical Validation of Models' Performance through Paired t-Test Analysis

Although the model's performance is evaluated using various metrics, the lack of comprehensive statistical significance testing prevents a rigorous assessment of whether the observed improvements are statistically significant or due to random chance.

Furthermore, a statistical t-test is conducted to confirm the significance of the performance improvements achieved by ASGD and CSGD, ensuring that the observed differences in performance are not due to random chance. This contribution is addressed in Section 4.8 (Statistical Validation of ASGD and CSGD Performance Enhancements via Paired t-Test Analysis), where a statistical t-test is conducted to validate that the performance improvements achieved by the ASGD and CSGD models are statistically significant and not due to random chance.

1.1.5. Enhancing Models' Interpretability and Explainability with Local Interpretable Model-agnostic Explanations and Shapley Additive Explanations

Finally, the fundamental research gap is the lack of XAI techniques in existing ETD systems. It is challenging for electricity companies to determine which consumer is detected as a possible fraud case because most detection models operate like black-boxes. For ML-based detection systems to become trusted by consumers and utilized in real-world environments, they must be interpretable.

However, interpretability and explainability are essential for ML-based detection systems to gain consumer confidence and be applied in real-world environments. However, in terms of interpretability and explainability, the framework incorporates advanced XAI techniques. Local Interpretable Model-agnostic Explanations (LIME) is used for interpretability, allowing us to examine how different features of the SGCC dataset influence the predictions of the ASGD and CSGD models locally with instance-level interpretations. In contrast, Shapley Additive Explanations (SHAP) globally provide deeper insights into feature relevance and the decision-making process of the models on the entire dataset. These contributions are discussed in Sections 4.9, 4.10, and 4.11 (LIME: Enhancing Interpretability of ASGD and CSGD Proposed Models, SHAP: Enhancing Explainability of ASGD and CSGD Proposed Models, and Critical Analysis of LIME and SHAP). These sections focus on the integration of LIME and SHAP techniques to provide interpretability and explainability to the ASGD and CSGD models, allowing stakeholders to better understand the decision-making process behind the models' predictions.

Conclusively, the implications of the AI-explained data-driven framework that integrates the ML algorithms ASGD and CSGD with XAI approaches (LIME and SHAP) have been thoroughly discussed in this study. The framework addresses critical challenges in ETD, including a lack of labeled data, class imbalance, and the difficulty in understanding the model's predictions. The proposed framework improves the accuracy and integrity of detecting fraudulent consumption patterns while ensuring that the decision-making process of the models is transparent, utilizing data-driven optimization and AL. This approach is particularly useful in overcoming the issues posed by the scarcity of labeled data and imbalanced datasets. The integration of XAI techniques such as LIME and SHAP allows for greater interpretability and explainability of the predictions made by the models. Overall, the proposed framework significantly enhances fraud detection accuracy, optimizes training parameters, and provides generalizability, explainability, and statistical significance. Moreover, the system is designed to be applicable in real-world environments, contributing to the practical application of ETD in modern smart grids.

2. Related Work

The related work section examines previous research and methodologies relevant to the proposed study, with a focus on advancements, challenges, and knowledge gaps in ETD. It is organized following the three general themes that entail the use of ML and DL in ETD, which brings into the center of attention in the development of these techniques and their usefulness in detecting fraudulent consumption, the use of optimization techniques on the model accuracy and efficiency, and the effects of AL in data efficiency, especially in the use of limited labeled datasets. This evaluation includes some thoughts about the strengths and limitations of existing methodologies and presents important areas where the work should be further developed.

2.1. Machine Learning and Deep Learning: Advancements in Electricity Theft Detection

In [9], the authors proposed ML-based techniques for ETD, employing the Synthetic Minority Over-sampling Technique-Tomeklinks (SMOTE-Tomek) to address class imbalance and ensure a more balanced dataset. They also used the feature extraction and scalable hypothesis (clean) algorithm to extract the most relevant features. The Cat Boosting (CB) algorithm was applied to determine whether a consumer is involved in theft. While the proposed model outperformed existing models with an accuracy of 93% and a detection rate of 92%, the limitation lies in its reliance on the SMOTE-Tomek technique, which may introduce synthetic noise that could affect the model's generalizability to real-world datasets.

Authors in [10] proposed an ensemble ML model that integrates multiple techniques into a single predictive model to reduce variance and bias. Their model, which combines Extreme Boosting (EB), Light Boosting (LB), Adaptive Boosting (AB), Random Forest (RF), CB, and Extra Trees (ET) algorithms, showed promising performance, with RF and ET achieving the highest Receiver Operating Characteristics Area Under the Curve (ROC-AUC) of 0.90. The ensemble model also outperformed others in terms of detection rates. However, the model's reliance on techniques like SMOTE and NearMiss undersampling to address class imbalance may lead to overfitting or an inaccurate representation of minority class samples in certain scenarios, particularly when applied to highly dynamic or evolving theft patterns.

The integration of advanced ML techniques into ETD and load forecasting has become a focal point in modern energy systems. In [11], the authors emphasized the use of models such as Decision Trees (DT), Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Logistic Regression (LR) for ETD, considering transformer reliability and supply interruptions. The study also highlights the potential direct impact of transformer failures on power revenue and the accuracy of theft detection. However, this method's dependence on aggregated time-series data means that it is unlikely to be sensitive to real-time fraud activity or individual consumer behavior, which could influence some detections.

An integrated model (DT, KNN, and SVM) was proposed by the authors in [13] to enhance the ETD. One-dimensional Wasserstein Generative Adversarial Networks (1D-WGAN) produced more authentic theft data, while the Kernel Fuzzy C-Means (KFCM) feature selection served as the model's foundation. Although the KFCM method, on which the model is built, may be sensitive to noisy and outlier data, this is the model's primary potential limitation. Additionally, fake data and resulting abnormal patterns could be produced by WGANs, which are then exploited to provide an inaccurate depiction of actual fraud activity.

The authors used a combination of Euclidean Convolutional Neural Networks (ECNN) and Graph Convolutional Networks (GCNN) to perform ETD with constraints on the basics used in [17]. GCNN resorts to graph convolutions to depict periodicity and time dependencies, whereas ECNN extracts hidden data characteristics. The hybrid model performed better when compared to the state-of-the-art models in ETD according to ROC-AUC and Mean Absolute Percentage Error (MAPE). Although it is effective, the limitation of this method is the complexity of a hybrid model, resulting in significantly problematic high costs in computation and increased processing time, especially where it is used with large data or real-time detection.

The authors in [18] proposed a Deep CNN (DCNN) model for efficient ETD in smart grids, analyzing consumption data from over 42,000 consumers over 24 months. The model implemented a random under-bagging strategy to handle imbalanced data and applied DCNNs on each subset. The model achieved an accuracy of 89%, but a key limitation is the random under-bagging strategy, which may not fully address the class imbalance, leading to underfitting in cases where fraudulent data is extremely sparse. Additionally, the model's performance in terms of precision and recall could be further improved to reduce False Positives (FP) and False Negatives (FN) in fraud detection.

2.2. Optimization Techniques: Improving Model Accuracy in Electricity Theft Detection Tuning the Hyperparameters

In [15], the authors proposed a Deep Neural Network (DNN) method for ETD using both frequency and temporal features. They applied Principal Component Analysis (PCA) and the Minimal Redundancy Maximum Relevance (MRMR) method for feature validation, and employed a Bayesian Optimization (BO) for hyperparameter tuning. The method achieved a 97% ROC-AUC and 91.8% accuracy, outperforming existing studies. However, the use of PCA may discard relevant information, affecting model interpretability and performance. Moreover, BO increases computational complexity, especially with large datasets.

In [16], the authors proposed an attention-based One-Dimensional CNN-Gated Recurrent Unit (1D-CNN-GRU) model optimized with Particle Swarm Optimization (PSO)

to enhance convergence and training efficiency for time-series forecasting. The model achieved high forecasting accuracy and used SHAP to improve transparency and interpretability. Federated Learning (FL) was employed to maintain data privacy by decentralizing model training. However, the integration of FL introduces challenges such as communication overhead and synchronization issues. Additionally, PSO, while efficient, can be computationally expensive for large datasets with numerous parameters.

In [19], the authors analyze hybrid load forecasting models, highlighting the role of intelligent systems such as DL and digital twins in optimizing power grid operations. They discuss the impact of factors like weather, economic status, and consumer behavior on forecasting accuracy, advocating for hybrid models to improve prediction reliability. Although the method has the advantage of increasing forecast accuracy, it has several drawbacks, including the difficulty of incorporating a large number of variables, which could increase computational costs and reduce model representation. Furthermore, when dealing with mixed and adaptive datasets, the use of hybrid models is linked to overfitting.

The authors in [20] provided a hybrid model, which was the combination of the Quantum Particle Swarm Optimization (QPSO) with Recurrent Neural Networks (RNN) to perform the load demand forecasting. The model utilized the QPSO algorithm to optimize RNN hyperparameters, where improved results were achieved in the forecasting performance of the educational buildings. Although the method showed considerable performance improvement, the limitation of the method is that QPSO is computationally intense, thus not ideal in situations where the information size is huge. Moreover, the dependence of the model on RNN can experience the problem of a vanishing gradient, and a lack of factors besides the occupancy and weather, data, which can be used to adjust the model, decreases its flexibility.

The authors in [21] enacted an Interpretable Generalized Additive Neural Network (IGANN) to be used in ETD computing on intelligent grids through well-balanced data and the efficient implementation of smart city grids. This approach integrates Reduced Noise (RN) with Synthetic Minority Oversampling Technique (SMOTE) data balancing, Reverse Lognormal Kalman Filter (RLKF) data preprocessing, and Multi-dimensional Spectral Graph Wavelet Transform (MSGVT) feature extraction capabilities. The IGANN model can be trained through the Secretary Bird Optimization Algorithm (SBOA) and can obtain an accuracy of 99%, which is better than the current method in terms of accuracy and recall. However, this strategy's weakness is that it relies on past data, which may not provide an accurate depiction of dynamic theft patterns. Also, the use of several techniques in the problem, along with SBOA optimization, makes them computationally complex and thus cannot be applied in real-time systems.

In [22], the authors proposed the Chaotic Harris Hawks Optimization (CHHO) algorithm for optimizing EV charging scheduling. The CHHO algorithm incorporates chaotic mapping to enhance exploration, improving the scheduling of EV charging and minimizing waiting times, travel costs, and energy usage. The model showed superior performance in terms of cost efficiency and charging time. However, a limitation of this method is its computational complexity, particularly when applied to large-scale EV fleets and charging stations. Additionally, while CHHO improves convergence speed, it still faces challenges in handling real-time dynamic conditions in large networks.

2.3. Active Learning Techniques: Improving Theft Detection Models' Efficiency Annotating the Datasets

In [12], the authors address the significant impact of ET on energy providers and smart grids, leading to both financial and non-technical losses. The study utilizes an energy dataset from the Open Energy Data Initiative, applying various ML techniques such as eXtreme Gradient Boosting (XGBoost), RF with AL, KNN with AL, Gradient Boosting (GB) with AL, DT with AL, and Boosting AL (BoostAL). The models were further improved with CB and Light Gradient Boosting Machines (LightGBM) classifiers. The proposed RF with AL model outperformed competing models, achieving improved accuracy. However, a limitation of this approach lies in the reliance on historical data, which cannot adequately account for evolving theft patterns. Additionally, the computational cost of applying multiple AL techniques can increase the training time and complexity.

In [14], the authors highlight the growing complexity of computer codes used in engineering problems, where calculating the probability of failure often requires time-consuming computations. The paper proposes an iterative method combining Kriging metamodels and Monte Carlo simulation for more efficient reliability assessment. This method, called AL-based Monte Carlo Simulation (AL-MCS), shows high accuracy in failure probability estimation with a reduced number of performance function calls. While the method demonstrates efficiency, a limitation is the computational cost associated with Kriging, which can still be substantial in high-dimensional or highly non-linear problems. Additionally, the accuracy of the results depends on the initial sampling, which could introduce bias if not appropriately handled.

In [23], the authors proposed an incremental learning and AL architecture for ETD, aimed at increasing adaptability while reducing labeling costs. The study introduced a Sample Filtering Strategy based on Gray Similarity and Kernel Functions (SFS-GSKF), which enhances the sample selection process, mitigating redundancy in data labeling. The proposed approach improves the model's capacity to learn new types of thieving behaviors by allowing for continual learning without experiencing catastrophic forgetting. However, this approach is constrained by the complexity

Table 2
Related work on electricity theft detection

Description	Methods Used	Limitation
Machine Learning and Deep Learning: Advancements in Electricity Theft Detection		
In [9], ML techniques used for ETD with a focus on SMOTE-Tomek for class imbalance.	SMOTE-Tomek, CB	Synthetic noise from SMOTE-Tomek can affect generalization.
In [10], an ensemble of ML techniques was proposed to improve predictive performance.	EB, LB, AB, RF, CB, ET	Reliance on SMOTE and NearMiss undersampling can lead to overfitting.
In [11], ML models like DT, SVM, KNN, and LR were used for ETD.	DT, SVM, KNN, LR	Limited to aggregated time-series data, unsuitable for real-time fraud detection.
In [13], an integrated model combining 1D-WGAN and KFCM was proposed for ETD.	1D-WGAN, KFCM	KFCM are sensitive to noise and outliers; potential issues with generated fake data.
In [17], a hybrid model integrating ECNN and GCNN was proposed to improve ETD.	ECNN, GCNN	High computational cost and complexity when used in large data or real-time systems.
In [18], DCNN was used to improve ETD by analyzing consumption data from over 42,000 consumers.	DCNN, Random Under-Bagging	Random under-bagging may not fully address class imbalance, leading to underfitting in sparse fraudulent data.
Optimization Techniques: Improving Model Accuracy in Electricity Theft Detection Tuning the Hyperparameters		
In [15], a DNN-based method with PCA and MRMR was proposed for ETD, optimizing model performance.	DNN, PCA, MRMR, BO	PCA may discard important data, reducing interpretability; BO increases computational complexity.
In [16], PSO-optimized CNN-GRU was proposed for forecasting with SHAP for interpretability.	PSO, CNN-GRU, SHAP, FL	PSO are computationally expensive for large datasets; FL introduces overhead and synchronization challenges.
In [19], hybrid models integrating DL and digital twins for load forecasting were proposed.	DL, Digital Twins	Difficulty in incorporating a large number of variables; potential for overfitting with mixed and adaptive datasets.
In [20], a hybrid model using QPSO with RNN was proposed for load forecasting.	QPSO, RNN	QPSO are computationally intensive; RNN faces the vanishing gradient problem.
In [21], IGANN was proposed for ETD with SBOA optimization, achieving high accuracy.	IGANN, SBOA, RN-SMOTE, RLKF	Relies on past data; SBOA optimization increases computational complexity.
In [22], the authors proposed the CHHO algorithm for optimizing EV charging scheduling.	CHHO	Computational complexity, difficulty handling real-time dynamic conditions in large networks.
Active Learning Techniques: Improving Theft Detection Models' Efficiency Annotating the Datasets		
In [12], the study applied AL to enhance ETD using multiple ML classifiers.	XGBoost, (RF, KNN, GB, DT, and Boost with AL)	Relies on historical data; computational cost increases with multiple AL techniques.
In [14], the method combines Kriging metamodels and Monte Carlo simulation for efficient reliability assessment.	Kriging Metamodels, MCS	Kriging are computationally costly in high-dimensional problems; initial sampling introduces bias.
In [23], an incremental learning and AL architecture for ETD with a Sample Filtering Strategy was proposed.	Incremental Learning, AL, SFS-GSKF	Computationally demanding sample filtering; biased performance depending on initial sampling data.
In [6], DSAN and CSAL were used for adapting models to new regions with different electricity consumption characteristics.	DSAN, CSAL	Relies on initial checks; large model adjustments required for dynamic adaptation.
In [24], an DAL method combining CNN with Monte Carlo dropout-based Bayesian active query was proposed.	CNN, Monte Carlo Dropout, Bayesian Active Query Selection	Ineffective in classes with high imbalances; computational complexity at a large scale.

of the sample filtering mechanism, which could be computationally demanding when dealing with large amounts of data. Additionally, the model may be biased because its performance may depend on whether the original sampling data was labeled.

The authors mention in [6] the issue of the decreased performance of data-driven detection models in the case of new regions in which the characteristics of the electricity consumption are different. The paper suggests the idea of using two stages, in which in Stage 1, a Deep Subdomain Adaptation Network (DSAN) is used to align the features of electricity consumption between different regions. In Stage 2, Core-Set Active Learning (CSAL) will identify samples that represent good choices to fine-tune the model with a minimum amount of resources. When the approach is efficient to adjust to new areas and enhance the rate of detection, its limitation is that this method is dependent on the availability of initial checks, which is not always possible in isolated areas. Also, the amount required to model the adjustment to adapt the model to regions may be huge when it comes to large implementations.

In [24], the authors discuss the expenses of manual labelling of a large amount of data in ETD. Their strategy is an intelligent Deep Active Learning (DAL), merging CNN with Monte Carlo dropout-based Bayesian active query, which is a data-driven, cost-efficient ETD solution. This strategy is highly effective in the sense that it will only provide useful training instances, maximizing the cost benefits: labeling costs will be decreased without a corresponding loss in model reliability. One limitation of the approach is that it works based on active query selection, which can be ineffective in classes with high imbalances. There may also be increased computational complexity that appears upon large scaling.

Despite the promising outcomes of traditional ML, DL, optimization, and AL methods in ETD, several challenges remain, including overfitting, data imbalance, high computational costs, and the need for effective hyperparameter optimization. These issues highlight the necessity for more resilient and adaptable models that can handle diverse datasets and real-world scenarios. Further research is required to develop models that are not only more efficient but also generalizable across various conditions, ensuring scalability and reliability in ETD frameworks. Addressing these gaps will pave the way for more robust and flexible ETD solutions. A summary of the related work is provided in Table 2.

3. Proposed Framework for Electricity Theft Detection

This section provides an overview of the workflow and structure of the proposed framework. The process starts with data collection, data preprocessing, where a mean imputer is used to handle missing values and guarantee data completeness, as given in Figure 2. Next, to tackle the problem of class imbalance and biased classification, LoRAS is employed, which creates synthetic instances while maintaining the geometric properties of the minority class. Later, ASGD is proposed, an AL-based model that chooses the most informative instances for annotation to handle unlabeled data. In contrast, we proposed CSGD, which improves parameter optimization and convergence by utilizing CS, as shown in Figure 3. Table 3 discusses the methodology used in this study.

3.1. Description of State Grid Corporation of China Dataset

In this study, data was collected from the SGCC dataset over 2 years and 1 month, from January 1st, 2014, to January 31st, 2016, with a total of 42,372 samples recorded during this timeframe [7, 25]. The dataset spans 1,035 days, with measurements taken at a daily resolution. It includes 1,036 features across the collected samples described in the main block ② of Figure 2. The data is provided in kiloWatt-hours (kWh), which is a common unit for energy measurement. These details provide valuable insights into the scope and structure of the SGCC dataset, which is particularly useful for analyzing electricity consumption patterns and detecting fraudulent activities within the dataset.

Figure 4 illustrates the electricity consumption (in kWh) across various dates, specifically spanning from January 2014 to September 2016. It highlights the variation in electricity usage, providing insights into trends over time of *Honest* and *Fraudulent* consumers. The *x-axis* of the plot represents the date, starting from January 2014 to September 2016, with the intervals marked as January, May, and September of each year. The *y-axis* shows electricity consumption in kWh.

Figure 4a shows a relatively steady consumption pattern, with two index readings at 6003 and 25009 that follow a consistent, smooth trend over the observation period. However, there is one notable exception: the consumption index reaches 39000 at a specific point in the data, creating a significant spike. This spike is the only fluctuation in the dataset and stands out from the otherwise stable readings. The reasons for this spike could be attributed to several factors, including a temporary increase in consumption due to external events, changes in consumer behavior, or even a special event that caused higher-than-usual electricity usage. After the spike at 39000, consumption returns to a steady level, indicating that the anomaly was brief and did not lead to lasting changes in consumption patterns. In contrast, Figure 4b shows three main consumption values for fraudulent consumers: 23000, 18505, and 25002. While the general trend follows a fluctuating pattern, the most prominent feature of the data is the presence of multiple spikes in consumption. These spikes, particularly around the index readings, indicate periods of unusually high electricity usage. The spikes might suggest potential fraudulent activities, such as illegal connections, unauthorized usage, or deliberate manipulation of consumption records. The presence of these spikes contrasts sharply with the relatively more stable consumption readings observed in honest consumers,

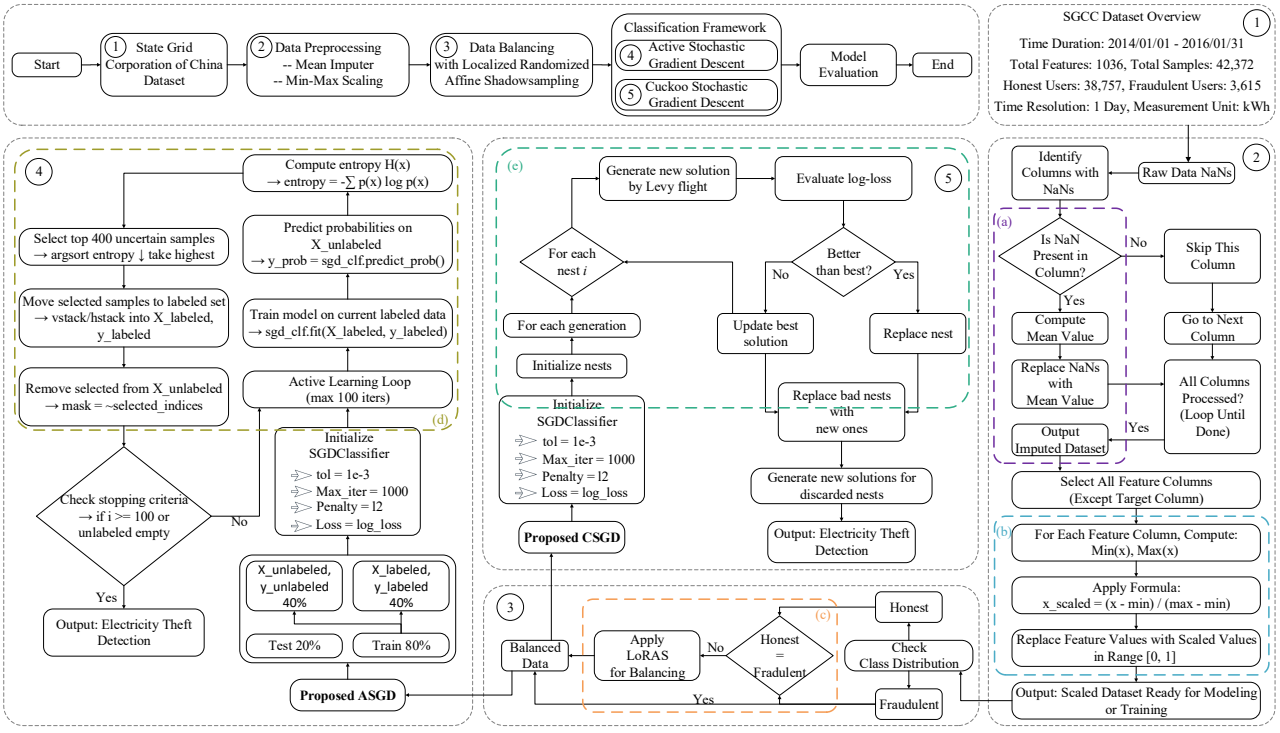


Figure 2: Flowchart illustrating the workflow process of the proposed framework for electricity theft detection using machine learning, active learning, and optimization techniques. It shows the step-by-step process of data preprocessing, balancing, preprocessing, model training, and electricity theft detection.

suggesting that fraudulent behavior is likely impacting the data.

However, the SGCC dataset consists of electricity consumption data used for detecting ET. The description of the SGCC dataset is outlined in Table 4.

Types of Electricity Theft: The dataset includes labels for instances of electricity consumption, classified into two categories: there are no specific types of theft in the dataset, only two classes:

- **Honest (Class 0)**: Instances representing legitimate electricity consumption.
- **Fraudulent (Class 1)**: Instances representing electricity theft, without distinguishing between specific types of theft.

Class Distribution and Imbalance Ratio: The class distribution in the dataset is highly imbalanced, with the majority of the instances belonging to the *Honest* class. Specifically, the distribution is as follows:

$$\text{Class 0 (Honest)} : 38,757 \text{ instances} \tag{1}$$

$$\text{Class 1 (Fraudulent)} : 3,615 \text{ instances} \tag{2}$$

The imbalance ratio between the two classes can be computed as the ratio of the *Honest* instances to the *Fraudulent* instances:

$$\text{Imbalance Ratio} = \frac{\text{Class 0 instances}}{\text{Class 1 instances}} = \frac{38,757}{3,615} \approx 10.73 \tag{3}$$

This indicates that the *Honest* instances outnumber the *Fraudulent* instances by a factor of approximately 10.73, which presents a significant challenge for efficient model training.

3.2. Dataset Preprocessing

In this section, we describe the preprocessing steps applied to the dataset to ensure its suitability for ML and DL model training. The steps include removing unnecessary features, handling missing values, and applying min-max scaling to the feature set.

Removing Irrelevant Features: The feature `CONS_NO` (which represents customer numbers or identifiers) is removed from the dataset because it does not contribute to the detection of ET [26]. This feature is an identifier that does not hold any predictive value for the model. The removal of this feature helps reduce dimensionality and prevents potential overfitting due to irrelevant features:

$$X_{\text{preprocessed}} = X \setminus \text{CONS\_NO} \tag{4}$$

where X is the original dataset containing all features and $X_{\text{preprocessed}}$ is the dataset after removing the `CONS_NO` column.

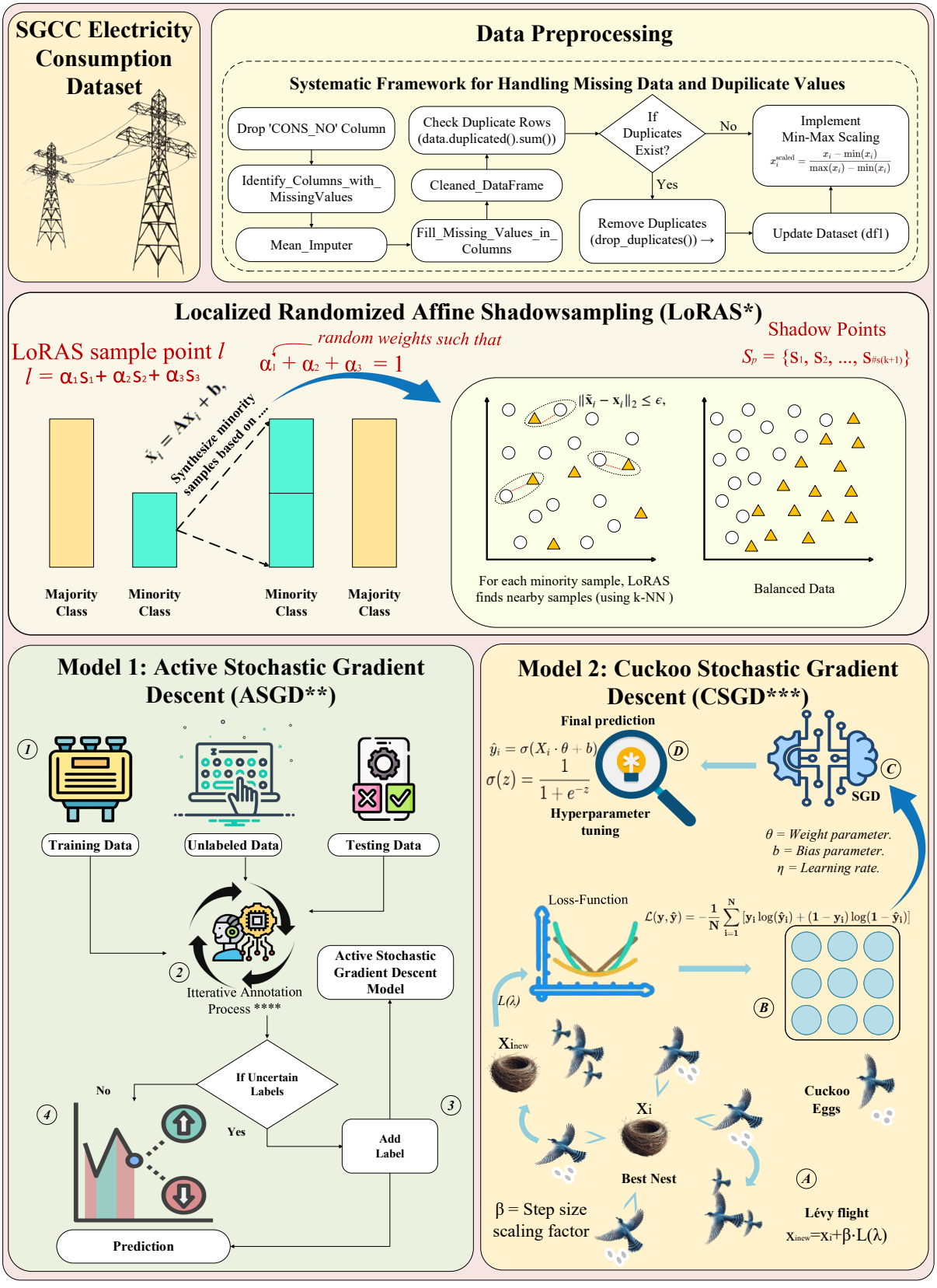


Figure 3: Illustration of the proposed AI-driven framework for electricity theft detection. * Helping ASGD and CSGD for unbiased theft detection by balancing the SGCC dataset with LoRAS is shown in Figure 5. ** The proposed ASGD model leverages entropy-driven sampling to accelerate accuracy is outlined in Algorithm 1. *** The proposed CSGD model utilizes Lévy flights for stochastic optimization improving gradient refinement and robustness is provided in Algorithm 2. **** The detailed iterative annotation process of active learning is shown in Figure 6. ***** The workflow of the proposed framework is visually presented in great detail in the flow diagram in Figure 2.

Table 3
Research methodology followed for this study

Technique	Purpose	Key Features	Applications
Removing Irrelevant Features	Removing features that do not contribute to the model's predictive power, like CONS_NO.	Eliminates noise, reduces dimensionality, and improves model performance.	Feature selection, data preprocessing.
Mean Imputer	Handle missing values by replacing them with the mean of the respective feature.	Preserves central tendency, prevents data loss.	Data preprocessing, handling incomplete datasets.
Min-Max Scaling	Rescale the features to a fixed range, usually [0, 1].	Scales features uniformly, helps in converging faster for gradient-based methods.	Feature scaling, normalization, data preprocessing.
LoRAS	Create artificial instances in the minority class to address class discrepancy.	Retains minority class geometry, and avoids overfitting.	Fraud detection, anomaly detection.
ASGD	Enhance learning by actively selecting high-uncertainty instances for labeling.	Entropy-based uncertainty sampling, efficient use of most informative data.	Semi-supervised learning, fraud detection.
CSGD	Combine CS optimization with SGD for better parameter optimization and handling complexities.	Global and local optimization, improved convergence.	Non-linear optimization problems, fraud detection.
10-FCV	Evaluate model generalization and performance on different subsets of data.	Minimizes overfitting and strikes a balance between variation and bias.	Model evaluation, result reliability testing.
SHAP	Evaluate each feature's role in the model's prediction.	Quantitative feature importance, global and local interpretability.	Decision-making, feature selection, and transparency.
LIME	Generate local explanations for individual predictions.	Instance-based explanations, perturbation analysis.	Model debugging, bias detection, and medical diagnosis.

Table 4
State Grid Corporation of China dataset description

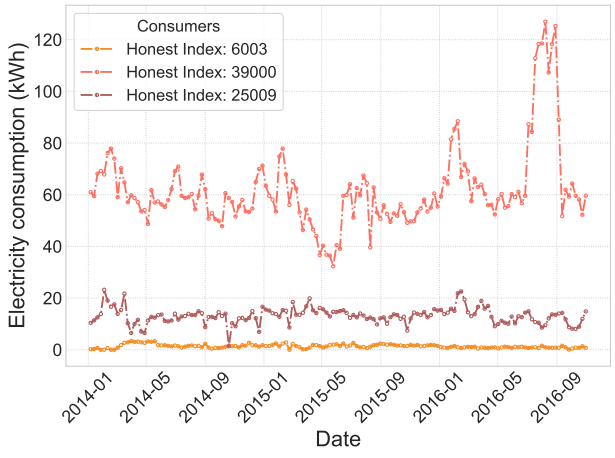
Description	Value
Time duration	2014/01/01 - 2016/1/31
Total samples	42372
Honest users	38757
Fraudulent users	3615
Fraudulent users percentage	8.53%
Days	1035
Time resolution	1 day
Unit	kWh

Handling Missing Values: Many real-world datasets contain missing values, which can be problematic for ML and DL models. In this study, missing values in the dataset are handled using mean imputation [27]. For each feature x_i, the missing values are replaced by the mean of the non-missing values in that feature as outlined in sub block *(a)* in main block ② of Figure 2. This is done to ensure that the dataset remains complete without introducing biases that could arise from removing rows with missing values. Mathematically, for each feature x_i in the dataset, the missing value is replaced by the mean of the existing values in that feature:

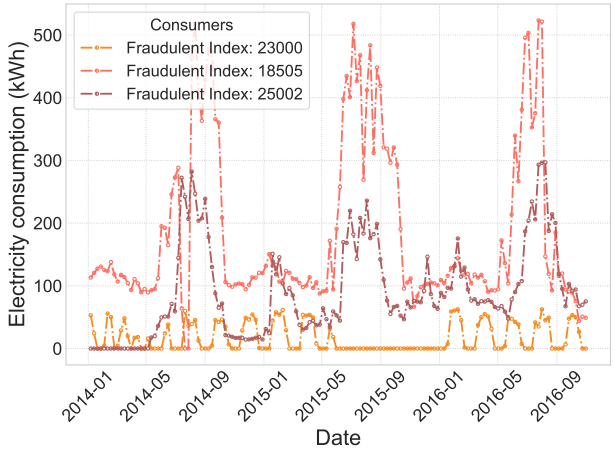
$$x_i^{\text{new}} = \begin{cases} x_i & \text{if } x_i \text{ is not missing} \\ \mu_i & \text{if } x_i \text{ is missing} \end{cases} \tag{5}$$

where x_i is the original value of feature x_i, x_i^{new} is the imputed value, $\mu_i = \frac{1}{n_i}\sum_{j=1}^{n_i} x_{ij}$ is the mean of feature x_i over the non-missing instances, and n_i is the number of non-missing values in feature x_i.

Min-Max Scaling: After handling missing values, the next step in preprocessing is the application of min-max scaling. This technique is used to standardize the range of features, making them fall within a fixed range of [0, 1]. This is crucial for ML and DL models that are sensitive to the scale of the features [7]. Mathematically, the min-max



(a) Honest consumers



(b) Fraudulent consumers

Figure 4: Illustration of honest and fraudulent consumers' readings in the SGCC dataset

scaling for each feature x_i is performed as follows:

$$x_i^{\text{scaled}} = \frac{x_i - \min(x_i)}{\max(x_i) - \min(x_i)} \tag{6}$$

where x_i is the original value of feature x_i, $\min(x_i)$ is the minimum value of feature x_i across all instances in the dataset, $\max(x_i)$ is the maximum value of feature x_i across all instances in the dataset, and x_i^{scaled} is the scaled value of feature x_i, which will lie within the range [0, 1] mentioned in sub block *(b)* in main block ② of Figure 2. The min-max scaling ensures that each feature contributes equally to the model, preventing features with larger ranges from dominating the learning process. This is particularly important in algorithms such as gradient descent, where features with larger values can cause the model to converge more slowly or unevenly.

After these preprocessing steps, the dataset is ready for the next stages, including splitting it into feature variables X and target variable y (*FLAG*), and further model training and evaluation.

3.3. Enhancing Data Representation in SGCC Dataset: Solving Imbalance Issues for ETD by LoRAS

LoRAS is an oversampling technique that creates synthetic instances for the minority class while maintaining its original distribution. The existing minority class data points undergo random affine transformations to generate synthetic instances, ensuring that the new data points remain within a localized region of the original feature space. This method preserves the geometric structure of the minority class while introducing controlled variability to enhance model generalization [28]. Figure 5 illustrates how LoRAS finds the minority class instances that need to be oversampled [29]. The method aims to generate synthetic data points by defining a localized region surrounding each minority class sample. To ensure that the generated instances do not significantly alter the distribution of the original data, this region is defined using a distance measure, such as the Euclidean distance. The synthetic sample $\tilde{\mathbf{x}}_i$ is created using the affine transformation:

$$\tilde{\mathbf{x}}_i = \mathbf{A}\mathbf{x}_i + \mathbf{b}, \tag{7}$$

where $\mathbf{x}_i$ represents an original minority class sample, $\mathbf{A}$ is a random linear transformation matrix that applies operations such as rotation, scaling, or shearing, and $\mathbf{b}$ is a random translation vector. The modifications preserve the minority class's structural properties while introducing controlled perturbations. To prevent adding too much variance, the newly created instances have to stay inside a restricted area [30]. This is achieved by ensuring that the Euclidean distance between the original and synthetic instances satisfies the condition:

$$\|\tilde{\mathbf{x}}_i - \mathbf{x}_i\|_2 \leq \epsilon, \tag{8}$$

where $\|\cdot\|_2$ denotes the Euclidean norm, $\tilde{\mathbf{x}}_i$ is the synthetic sample, $\mathbf{x}_i$ is the original minority class sample, and $\epsilon > 0$ is a hyperparameter that defines the maximum allowable perturbation. Because of this constraint, the generated instances remain sufficiently close to the original distribution while introducing diversity. Especially with high-dimensional or imbalanced datasets, LoRAS introduces controlled perturbations into the feature space, increasing robustness and successfully reducing overfitting given in sub block *(c)* in the main block ③ of Figure 2.

In ETD, where imbalanced datasets are a frequent problem, LoRAS is particularly effective in addressing the challenge of theft detection in ETD, where fraudulent activities often constitute only a small fraction of the overall dataset. LoRAS improves the model's capacity to differentiate between honest and fraudulent consumers by producing synthetic instances that maintain the geometric characteristics of fraudulent instances. As a result, fraud identification becomes more trustworthy as detection rates rise. Comparably, LoRAS aids in reducing class imbalance in ETD, guaranteeing that the model learns fraudulent consumers efficiently without overfitting to the majority class.

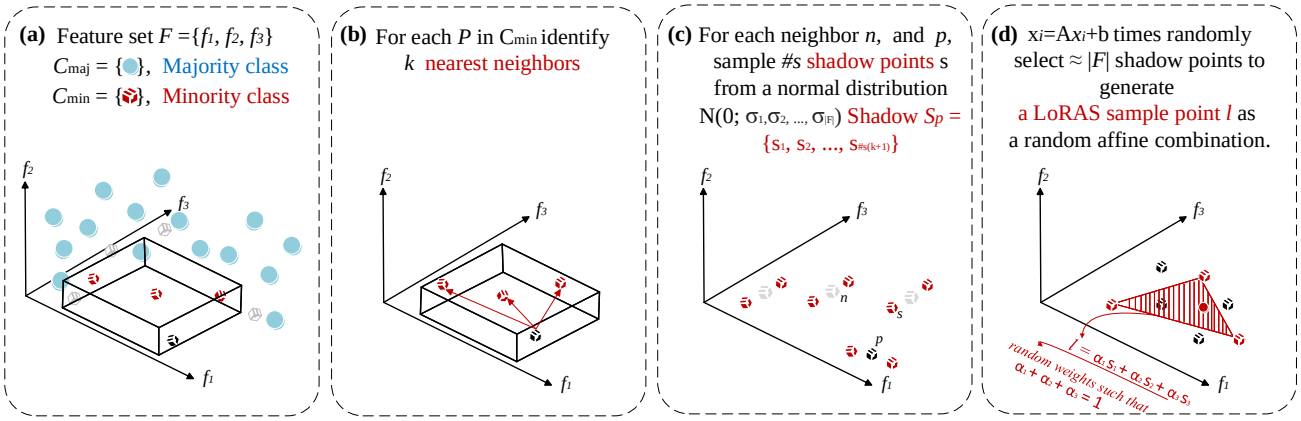


Figure 5: Working of localized randomized affine shadowsampling (LoRAS) for synthetic data generation. The functioning of LoRAS in the proposed framework is shown in Figure 3, where it takes preprocessed SGCC dataset from the upper block and delivers the balanced data to Model 1 (ASGD) and Model 2 (CSGD) for unbiased theft detection.

3.4. Towards Efficient Electricity Theft Detection with Newly Proposed ASGD and CSGD

This study proposes two novel models, ASGD and CSGD, to improve classification accuracy and enhance ETD in the SGCC dataset. These models are designed to refine model parameters and focus on the most informative instances, ensuring better performance and robustness in the learning process. By integrating metaheuristic optimization and AL, the models prioritize informative instances and optimize parameters, thereby enhancing classification and robustness against noisy and imbalanced data in ETD. Unlike conventional SGD models, the ASGD model incorporates entropy-based AL which strategically focuses learning on the most uncertain data points. This innovation improves data efficiency and enhances performance in settings where labeled data is scarce. Furthermore, the CSGD model introduces a novel integration of CS metaheuristics with SGD, enabling dynamic and global hyperparameter optimization. This approach mitigates the sensitivity of SGD to learning rate and ensures faster, more stable convergence in the presence of class imbalance. These models address critical challenges in ETD that existing techniques often overlook, including poor generalization, optimization instability, and limited interpretability.

3.4.1. Stochastic Gradient Descent (SGD): Formulation and Objective

SGD is a widely used optimization technique in ML, particularly for training models such as LR. In this context, SGD is used to iteratively update the model parameters by computing the gradient of the loss function with respect to the model's parameters, based on individual data points or small batches. This contrasts with traditional gradient descent, which computes the gradient over the entire dataset. The advantage of SGD lies in its computational efficiency, especially when dealing with large-scale datasets, as it drastically reduces the computational complexity by avoiding the need to process the entire dataset at once. Given a dataset $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, where $\mathbf{x}_i \in \mathbb{R}^d$ represents the feature vector and $y_i \in \{0, 1\}$ denotes the binary class label, the objective of SGD is to find an optimal weight vector $\mathbf{w}$ that minimizes a predefined loss function. The optimization problem can be expressed as:

$$\min_{\mathbf{w}} \quad \mathcal{L}(\mathbf{w}) + \lambda R(\mathbf{w}), \tag{9}$$

where $\mathcal{L}(\mathbf{w})$ represents the empirical loss function, $R(\mathbf{w})$ is the regularization term, and λ is the regularization coefficient that controls the trade-off between model complexity and generalization. For LR, the loss function is given by the log-loss function:

$$\mathcal{L}(\mathbf{w}) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log \sigma(\mathbf{w}^T\mathbf{x}_i) + (1-y_i)\log(1-\sigma(\mathbf{w}^T\mathbf{x}_i))\right], \tag{10}$$

where $\sigma(z) = \frac{1}{1+e^{-z}}$ is the sigmoid activation function, which maps the linear combination of features to a probability. In the SGD optimization process, the weight vector $\mathbf{w}$ is updated iteratively using the following update rule:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta\left(\nabla\mathcal{L}(\mathbf{w}^{(t)}) + \lambda\nabla R(\mathbf{w}^{(t)})\right), \tag{11}$$

where η is the learning rate, and $\nabla\mathcal{L}(\mathbf{w})$ represents the gradient of the log-loss function with respect to the weight vector $\mathbf{w}$.

While SGD offers significant advantages in terms of efficiency, it also faces several challenges, particularly when applied to complex tasks such as ETD. These challenges include sensitivity to the learning rate, poor generalization on noisy or imbalanced datasets, and optimization instability. To address these issues, the proposed ASGD and CSGD models build upon the principles of SGD, incorporating novel techniques that enhance model performance and stability, especially in large, imbalanced, and noisy datasets.

3.4.2. Rationale Behind Innovation of ASGD and CSGD

The selection of ASGD and CSGD is driven by the unique challenges inherent in ETD, including limited labeled data and the need for efficient and interpretable models. ASGD integrates entropy-based AL with SGD, allowing the model to intelligently and iteratively select the most uncertain data samples for labeling and training. This not only reduces annotation effort but also improves learning efficiency by focusing on informative instances. This approach contrasts with conventional passive learning, which uniformly treats all instances and wastes computational resources on less relevant data. On the other hand, CSGD is designed to optimize the training process by integrating CS with SGD. This combination enables automatic and dynamic tuning of critical hyperparameters such as learning rate. CS provides a powerful global optimization mechanism that enhances convergence stability and avoids local minima, issues often encountered with manual or grid-based parameter tuning in stochastic optimization.

Both ASGD and CSGD extend beyond simple adaptations of existing methods. They are tailored to address specific gaps identified in Section 1.1, namely, the inefficiency of traditional SGD in imbalanced and partially labeled datasets, and the need for robust optimization strategies. These innovations are further validated through statistical significance testing and performance comparisons with baseline models.

3.4.3. Model 1: Proposed ASGD Model for Efficient Electricity Theft Detection: Leveraging Entropy-Driven Sampling Mechanism

SGD often wastes computational resources by processing data points that are redundant or less informative, due to its lack of an effective mechanism for identifying valuable training instances. Entropy-based AL iteratively selects the most ambiguous data points for labeling data points to be used for training, which overcomes the drawbacks of SGD. Entropy-based AL prioritizes instances that show the highest level of uncertainty rather than treating all instances equally [33]. This significantly reduces the number of labeled samples required while simultaneously improving model performance, making it particularly beneficial in ETD, where labeled data is expensive. The following steps make up the AL process, as shown in Figure 6. *a)* The dataset is split into labeled and unlabeled parts, with only a small part initially labeled, to ensure that the SGD depends on AL for improvement. *b)* To improve decision boundaries based on labeled instances, weights and biases are iteratively updated in SGD training, which is used to evaluate the ASGD model. *c)* After training, the model evaluates unlabeled data and identifies high-uncertainty instances located near decision boundaries for annotation. *d)* The selected instances are labeled by the SGD model, reducing manual annotation costs by concentrating only on the most instructive instances. *e)* Through repeated retraining with newly labeled data, the model improves predictions and classification accuracy across multiple iterations. We describe the ASGD model in several key components, as seen in Figure 3.

① The ASGD model uses entropy-based uncertainty sampling to integrate AL with the traditional SGD model. By prioritizing instances that contribute the most to the learning process, this approach enables efficient model training even with progressively expanding labeled datasets.

② The AL process is initiated, and the model iterates through 100 rounds to identify the most uncertain instances from the unlabeled dataset (D_U). At each iteration, the classifier predicts probability estimates:

$$\hat{p}(x_j) = [p_0(x_j), p_1(x_j)], \tag{12}$$

where x_j represents a data point or feature vector corresponding to the j-th instance in the dataset. The term $\hat{p}(x_j)$ denotes the predicted probability distribution over the possible classes for the given input x_j. The uncertainty of each prediction is calculated using Shannon entropy:

$$H(x_j) = -\sum_{k\in\{0,1\}} p_k(x_j) \log p_k(x_j), \tag{13}$$

where $H(x_j)$ represents the entropy of the predicted probability distribution for the instance x_j. The summation iterates over the possible classes, where $k \in \{0, 1\}$, indicating a binary classification problem. The term $p_k(x_j)$ denotes the predicted probability that the instance x_j belongs to class k, where $k = 0$ corresponds to one class and $k = 1$ corresponds to the other class. The logarithm function $\log p_k(x_j)$ is applied to these probabilities to compute the entropy. The negative summation ensures that entropy is always non-negative, with higher values indicating greater uncertainty in classification. A low entropy suggests a confident prediction, whereas a high entropy implies that the model is uncertain about the classification of x_j. Therefore, instances with higher entropy are considered more valuable for labeling, as they are expected to contribute more to the learning process. The top B most uncertain instances are selected using:

$$S = \arg\max_{|S|=B} H(x_j), \quad \forall x_j \in D_U, \tag{14}$$

where S represents the selected subset of data points from the unlabeled dataset D_U. The goal is to choose S such that it contains the most informative instances for labeling. The instances are then transformed from D_U unlabeled to D_L labeled. The classifier is retrained on the updated labeled dataset, refining its decision boundary.

③ By choosing the most uncertain instances, the AL process iterates, improving the model's decision boundary with each cycle. The classifier is retrained on the updated labeled set D_L after the labeling of the top B uncertain examples. To ensure that the model gives priority to the most informative data, this process is repeated until all instances are correctly labeled or 100 iterations are completed mentioned in sub block *(d)* in the main block ④ of Figure 2.

④ Unlike classic passive learning, which trains models

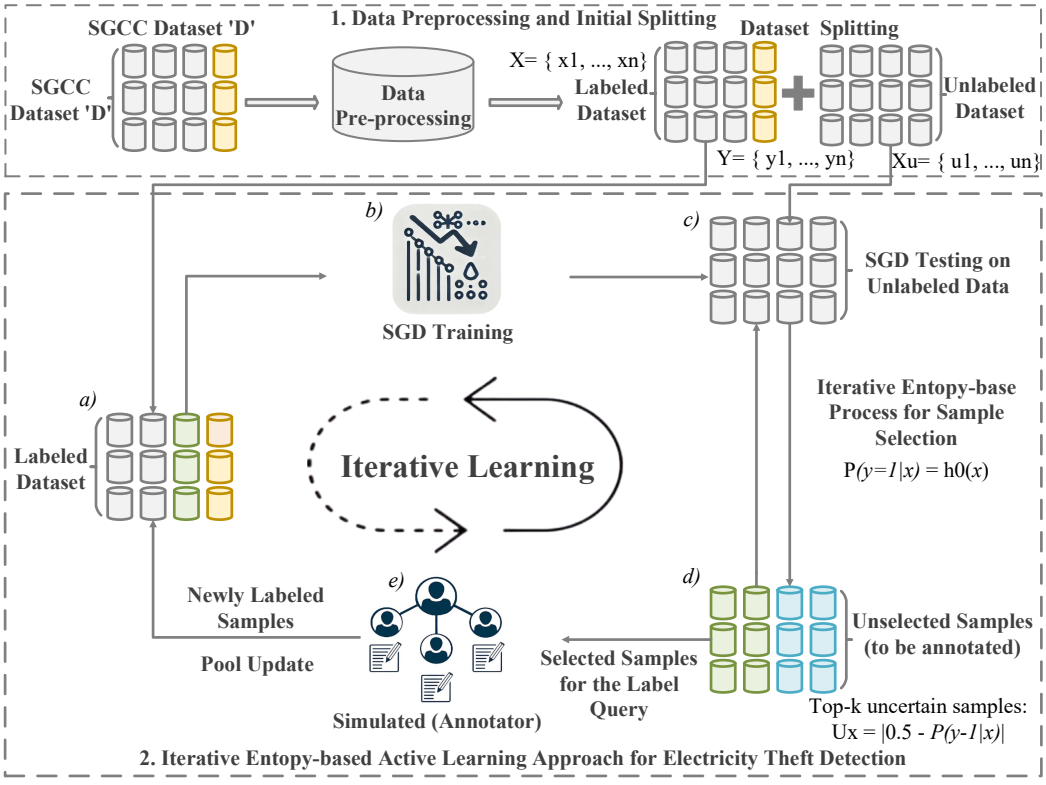


Figure 6: Enhancing SGD efficiency in electricity theft detection through entropy sampling and active learning annotation in ASGD. The detailed workflow of the ASGD process, including entropy sampling and active learning annotation, is illustrated in Figure 3.

using static datasets, the ASGD technique leverages AL-driven sample selection to adaptively modify its decision-making process. In domains where labeled data is expensive or scarce, this is particularly beneficial. Due to the entropy-based AL process, the model maintains high classification performance while minimizing manual labeling effort. The main steps affecting the computational complexity of the ASGD model are the entropy computation and sample selection processes. Even though typical SGD updates have a linear complexity, an additional entropy-based selection step adds a small computational burden. However, the benefits of improving learning efficacy and reducing annotation costs balance this expense. Iterative refinement of decision boundaries ensures that the model gradually gets closer to optimal performance with the least amount of labeled input. Overall, the uncertainty-driven AL framework enhances model generalization, demonstrating its practicality in real-world applications that require efficient data annotation and classification. The workflow of ASGD is given in Algorithm 1 and Figure 3. The algorithm integrates SGD with AL to detect ET. It divides the dataset into labeled, unlabeled, and test sets and initializes the model's parameters, such as weights θ and bias b. Weights are iteratively updated by computing prediction errors based on sigmoid-based probability outputs, after the model is trained on the labeled data using SGD in each AL iteration. Following training, entropy-based uncertainty sampling computes entropy over the predicted probabilities of the samples to identify the most uncertain ones from the unlabeled set. Iteratively, the training data is expanded by adding the top $b\%$ of these high-entropy samples to the labeled set. By selecting the most informative samples, this procedure is repeated for a predetermined number of AL iterations, progressively improving the model.

3.4.4. Model 2: Proposed CSGD Model for Electricity Theft Detection: Utilizing Lévy Flights for Stochastic Optimization

SGD is sensitive to hyperparameters, especially the learning rate. Selecting an unsuitable learning rate can result in slow convergence or divergence, which makes model training difficult. A very low learning rate leads to prolonged training with suboptimal convergence, while an excessively high learning rate causes the optimization process to overshoot the ideal solution. For optimal learning rate selection, we applied the CS algorithm to address the sensitivity of SGD hyperparameters, specifically the learning rate. Inspired by cuckoo bird brood parasitism, CS is a metaheuristic optimization algorithm that uses elitist selection mechanisms and Lévy flight-based random walks to choose the best solutions. Using CS, we methodically investigate and utilize the search space to find the ideal learning rate that strikes a balance between stability and convergence speed.

Algorithm 1 Proposed Active Stochastic Gradient Descent (ASGD) for Electricity Theft Detection: The algorithm's working is depicted in Figure 3.

Input: Initial weights θ, bias $b = 0$, learning rate η, number of SGD iterations *n_iterations*, number of active learning iterations *n_active_iterations*, labeled data $X_{\text{labeled}}, y_{\text{labeled}}$, unlabeled data $X_{\text{unlabeled}}$, and test data $X_{\text{test}}, y_{\text{test}}$
Output: Trained ASGD model, evaluation metrics
Initialization:

- Randomly initialize θ and set $b = 0$
- Split dataset into labeled $X_{\text{labeled}}, y_{\text{labeled}}$, unlabeled $X_{\text{unlabeled}}$, and test set $X_{\text{test}}, y_{\text{test}}$

for $i = 1$ **to** *n_active_iterations* **do**
    Train model with SGD on labeled data:
    for $e = 1$ **to** *n_iterations* **do**
        for each sample (x_j, y_j) in X_{labeled} **do**
            Compute predicted probability $\hat{y}_j = \sigma(x_j^T\theta + b)$ where σ is the sigmoid function
            Compute error error $= \hat{y}_j - y_j$
            Update weights and bias:

$$\theta := \theta - \eta \cdot \text{error} \cdot x_j, \quad b := b - \eta \cdot \text{error}$$

        end for
    end for
    Entropy-based Uncertainty Sampling:
    for each sample x_k in $X_{\text{unlabeled}}$ **do**
        Compute predicted probability $p_k = \sigma(x_k^T\theta + b)$
        Compute entropy:

$$H(p_k) = -p_k \log(p_k) - (1 - p_k)\log(1 - p_k)$$

    end for
    Select the top $b\%$ instances with the highest entropy from $X_{\text{unlabeled}}$ and add them to the labeled data $X_{\text{labeled}}, y_{\text{labeled}}$
end for
Model Evaluation: To evaluate the model, accuracy was assessed using X_{test}, y_{test} and presented in Table 7. Precision, recall, and F1-score are depicted in Figure 8, while the ROC-AUC is shown in Figure 9, and the PR-AUC is illustrated in Figure 10.

Algorithm 2 Proposed Cuckoo Stochastic Gradient Descent (CSGD) for Electricity Theft Detection: The algorithm's working is visualized in Figure 3.

Input: Training data (X, y), Learning rate η, Cuckoos N, Lévy parameter λ, Max iterations T
Output: Optimized parameters θ and b
Initialize nests $x_1, \ldots, x_N$ randomly.
Define log-loss:

$$\mathcal{L}(y, \hat{y}) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)\right]$$

for each iteration t **do**
    Update nests using Lévy flight and evaluate log-loss.
    Replace nests based on fitness.
    Retain best solutions and refine with SGD.
end for
Fine-tune parameters using SGD:
    Compute gradients:

$$\frac{\partial L}{\partial \theta} = X^T(\hat{y} - y), \quad \frac{\partial L}{\partial b} = \sum_{i=1}^{N}(\hat{y}_i - y_i)$$

    Update parameters:

$$\theta \leftarrow \theta - \eta \cdot \frac{\partial L}{\partial \theta}, \quad b \leftarrow b - \eta \cdot \frac{\partial L}{\partial b}$$

Compute final predictions with sigmoid:

$$\hat{y}_i = \sigma(X_i \cdot \theta + b)$$

Apply threshold:

$$\hat{y}_i = \begin{cases} 1 & \text{if } \hat{y}_i \geq 0.5 \\ 0 & \text{if } \hat{y}_i < 0.5 \end{cases}$$

Model Evaluation: To evaluate the model, accuracy was assessed using X_{test}, y_{test} and presented in Table 7. Precision, recall, and F1-score are depicted in Figure 8, while the ROC-AUC is shown in Figure 9, and the PR-AUC is illustrated in Figure 10.

Ⓐ The CSGD model enhances binary classification tasks by combining CS optimization with SGD. This combination achieves superior parameter optimization by leveraging SGD's local refinement and CS global search capabilities [34]. The CS optimization technique employs Lévy flights to generate new solutions or nests based on the behavior of cuckoos, which lay their eggs in other birds' nests [35]. CS aims to minimize a log-loss function given in sub block *(e)* in the main block ⑤ of Figure 2, which is defined for binary classification:

$$\mathcal{L}(y, \hat{y}) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)\right], \quad (15)$$

where y_i is the actual label, $\hat{y}_i$ is the predicted probability and N is the number of instances. The CS algorithm uses Lévy flights to generate new solutions and aims to minimize the loss by exploring the search space.

$$x_i^{\text{new}} = x_i + \beta \cdot L(\lambda) \quad (16)$$

The update rule for a nest x_i, with the scaling factor β acting as a guide for the adjustment. By capturing a dynamic modification that can be generated via gradients, regularization, or stochastic perturbations, the transformation $L(\lambda)$ guarantees controlled and adaptive feature refinement in optimization processes. Ⓑ The model employs SGD to further fine-tune the parameters following CS optimization.

The iterative process of updating the weight and bias is carried out. In this case, the update step size is determined by the learning rate n, and the model parameters being improved are θ and b. The terms $\frac{\partial L}{\partial b}$ and $\frac{\partial L}{\partial \theta}$ represent the gradients of the loss function L with regard to b and θ, respectively. The model adjusts these parameters in the direction of the negative gradient to minimize the loss function, thereby ensuring effective convergence toward an optimal solution.

$$\theta \leftarrow \theta - \eta \cdot \frac{\partial L}{\partial \theta} b \leftarrow b - \eta \cdot \frac{\partial L}{\partial b} \tag{17}$$

where η is the learning rate, and the gradients $\frac{\partial L}{\partial \theta}$ and $\frac{\partial L}{\partial b}$.
Ⓒ The model's predictions are computed using the σ *sigmoid* activation function, applied to the linear transformation of input features X_i with weights θ and bias b to define the expected output $\hat{y}_i$. In neural networks and LR, this formulation is essential for converting inputs into probabilistic outputs in binary classification tasks.

$$\hat{y}_i = \sigma(X_i \cdot \theta + b) \tag{18}$$

In the binary classification decision rule, the predicted output $\hat{y}_i$ is assigned a label of 1 if its probability estimate is ≥ 0.5, and a label of 0 otherwise, LR and neural networks depend on this thresholding method to enable discrete class predictions from continuous *sigmoid* activated probabilities.

$$\hat{y}_i = \begin{cases} 1 & \text{if } \hat{y}_i \geq 0.5 \\ 0 & \text{if } \hat{y}_i < 0.5 \end{cases} \tag{19}$$

Ⓓ Metrics such as accuracy, precision, recall, F1-score, and ROC-AUC are used to evaluate the CSGD model's performance and offer an extensive assessment of the classification's performance. The working of the CSGD model is outlined in Algorithm 2 and Figure 3. To detect ET, the CSGD algorithm combines SGD and CS optimization. Initially, a log-loss function is defined for model evaluation, and a population of candidate solutions (nests) is randomly initialized. Using Lévy flight, CS updates these nests in each iteration, assesses their fitness using log-loss, and swaps out weaker solutions for more promising ones. The optimal nests are kept and improved upon through CSGD, which iteratively updates weights and bias by computing gradients of the loss function for the model parameters. After optimization, a *sigmoid* activation function is applied to generate probability scores, followed by a thresholding operation to classify instances as either theft or non-theft.

4. Results and Discussion

The classification performance of the proposed models, ASGD and CSGD, is evaluated using confusion matrices and other performance metrics. By displaying the quantity of True Positives (TPs), FPs, True Negatives (TNs), and FNs, the confusion matrices offer a thorough analysis of the model's capacity to differentiate between cases that are fraudulent and those that are honest. In ETD, the models' performance is assessed using several performance metrics given in Table 5. Accuracy is the percentage of correctly classified cases as demonstrated in Table 5 (equation 1), and precision is the percentage of TP predictions among all positive predictions, as shown in Table 5 (equation 2). Recall the model's sensitivity to detecting positive instances by correctly identifying real positive cases as illustrated in Table 5 (equation 3). The F1-score is a metric that balances precision and recall, Table 5 (equation 4). Table 6 (equation 5) shows the performance insights for both the base model and the proposed model. The ROC-AUC integrates the True Positive Rate (TPR) over the False Positive Rate (FPR) to assess a model's capacity for class distinction.

Table 5
Performance metrics used in this work

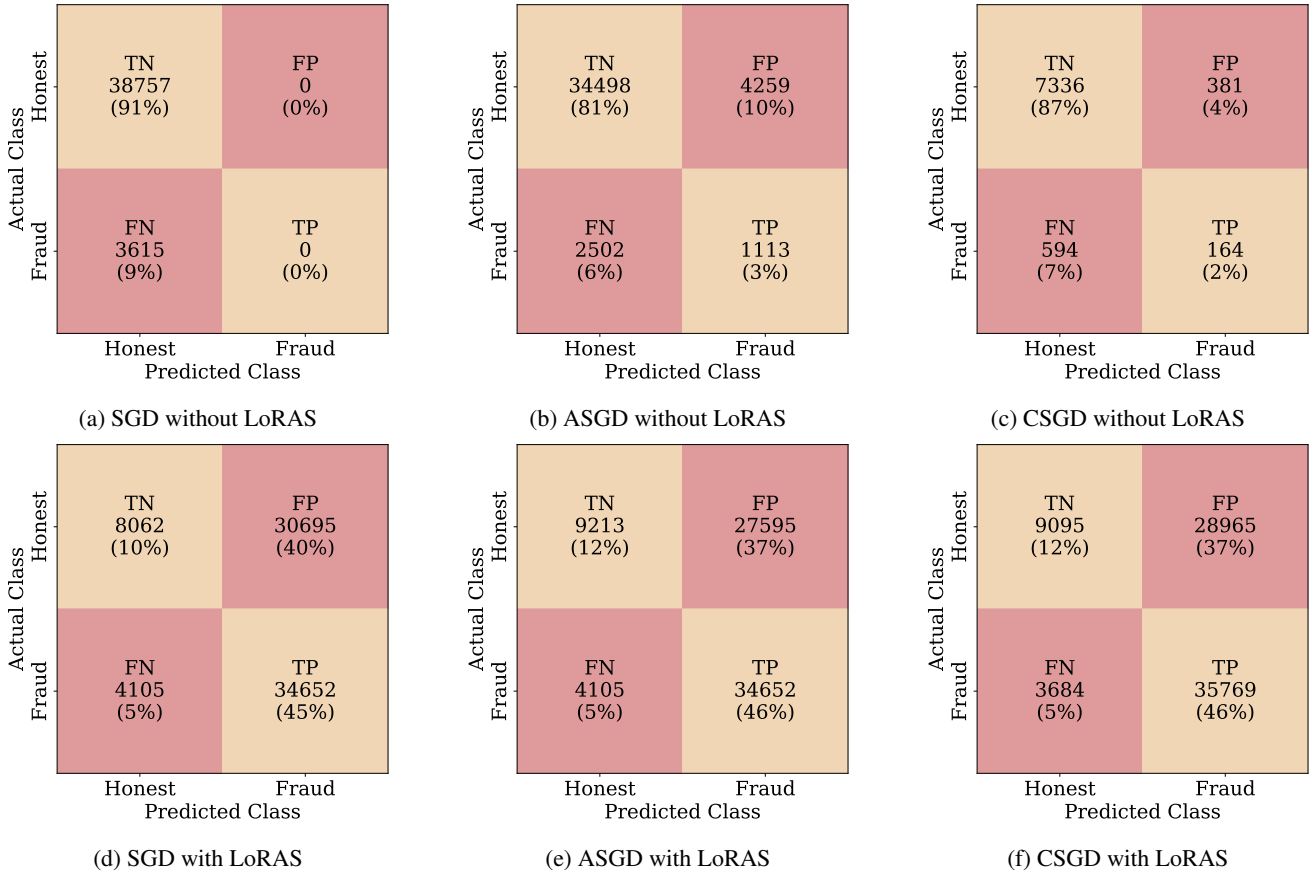
Metric	Formula
Accuracy	$\frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$
Precision	$\frac{\text{TP}}{\text{TP} + \text{FP}}$
Recall	$\frac{\text{TP}}{\text{TP} + \text{FN}}$
F1-Score	$2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$
ROC-AUC	$\int_{-\infty}^{\infty} \text{TPR}(t)\, d\text{FPR}(t)$
PR-AUC	$\int_{0}^{1} \text{Precision}(t)\, d\text{Recall}(t)$

4.1. Data Balancing Performance Analysis

To assess the efficacy of the SGD, ASGD, and CSGD models in identifying fraudulent cases, we compare their performance with and without the LoRAS balancing technique. Our goal is to evaluate the effect of data balancing on fraud detection accuracy and overall model performance by comparing the classification outcomes of the models. Each model responds differently to class imbalances, especially when it comes to differentiating between fraudulent and non-fraudulent users, as revealed by the respective confusion matrices. The confusion matrices for SGD, ASGD, and CSGD models, both with and without the LoRAS balancing technique, are presented in Figure 7, offering a comprehensive analysis of each model's classification performance. Without LoRAS, the standard SGD model (Figure 7a) struggles significantly in detecting fraudulent cases, recording 0 TPs and has a high number of 38757 TNs. The large count of 3615 FNs indicates that many fraudulent instances were misclassified as honest users, while the 0 FPs show no honest users were incorrectly labeled as fraudulent. This highlights the model's inability to effectively distinguish between fraudulent and honest cases, heavily favoring

Table 6
Performance and insights of SGD, ASGD and CSGD for electricity theft detection

Model/ Accuracy	Insights
SGD 0.60	SGD uses individual data instances to iteratively update model parameters, acting as a baseline optimization technique. It performs poorly in classification and has trouble with unbalanced datasets, which results in weak decision boundaries. Its inability to prioritize informative instances due to a lack of adaptive learning mechanisms reduces its effectiveness when handling challenging fraud detection tasks.
ASGD 0.82 (36.67% improvement over SGD)	Utilizes AL techniques alongside SGD, actively selecting and prioritizing instances with high uncertainty. This approach helps refine decision boundaries and enhances the model's ability to generalize, particularly for imbalanced datasets. Achieves the best results.
CSGD 0.81 (35% improvement over SGD)	Combines CS with SGD to leverage global optimization capabilities. The model can explore a broader parameter space, reducing the chances of getting trapped in local minima. This results in improved accuracy, although it requires higher computational resources.



(a) SGD without LoRAS (b) ASGD without LoRAS (c) CSGD without LoRAS

(d) SGD with LoRAS (e) ASGD with LoRAS (f) CSGD with LoRAS

Figure 7: Confusion matrices for SGD, ASGD, and CSGD with and without LoRAS balancing

honest classifications and failing to detect fraud. ASGD, without LoRAS (Figure 7b), demonstrates a slight improvement over SGD. The model increases its TPs to 1113 and reduces FNs to 2502, suggesting better fraud detection. However, this comes at the cost of an increased FP count of 4259, indicating more honest users being misclassified as fraudulent. Despite this, the model maintains a strong TN count of 34498, accurately classifying most honest users. The improved TP count suggests that ASGD's averaging mechanism stabilizes gradient updates, enabling the model to better capture fraud patterns, though at the expense of a higher FPR. CSGD without LoRAS (Figure 7c) further improves fraud detection, achieving 164 TPs and reducing FNs to 594. Furthermore, 7336 instances are accurately classified as TNs by the model, with 381 FPs, which indicate that some honest users were mistakenly identified as fraudulent users. A better balance between detecting fraud and minimizing misclassification errors is suggested by the

model's relatively lower number of FPs when compared to other configurations, even though it successfully captures more fraudulent cases. While the model becomes more effective at identifying fraudulent cases, it also misclassifies more legitimate users as fraudulent, suggesting a potential overfitting to fraud patterns despite its optimization benefits. The application of the LoRAS data balancing technique results in notable improvements across all models. With LoRAS, SGD (Figure 7d) significantly increases its fraud detection capability, achieving 34652 TPs while reducing FNs to 4105. However, this comes with a trade-off, as the FP count spikes to 30695, and TNs drop to 8062, indicating that many honest users are misclassified as fraudulent. ASGD with LoRAS (Figure 7e) achieves a more balanced performance, with 34652 TPs, 4105 FNs, and an improved TN count of 9213, though it still suffers from a high FP count of 27595. This suggests that while ASGD with LoRAS detects fraud effectively, it still struggles to distinguish between fraudulent and honest users. CSGD with LoRAS (Figure 7f) demonstrates the most balanced performance among the three models. It achieves 35769 TPs, reduces FNs to 3684, and improves honest user classification with 9095 TNs while lowering FPs to 28965. This indicates that CSGD, when combined with LoRAS, achieves a better trade-off between detecting fraudulent cases and correctly identifying honest users, effectively leveraging its controlled stochastic updates to refine classification boundaries and reduce misclassifications.

4.2. Justification for Benchmark and Proposed Models

In ETD, modeling is complicated by high-dimensional data, extreme class imbalance, and the need for scalability and interpretability. We select baseline models that are widely used in ETD literature, KNN [9, 11, 12], SVM [11, 13, 14], LR [11], SGD [36], TabNet [37], and DL models like CNN-GRU [16]. These models cover a range of traditional approaches. We justify our proposed ASGD and CSGD models by directly addressing limitations observed in these benchmarks.

- KNN is a non-parametric method that compares each test instance with all training data. While effective in low-dimensional tasks, KNN scales poorly ($\mathcal{O}(n \times d)$), is sensitive to irrelevant features, and lacks model generalization in high-dimensional datasets like SGCC. However, ASGD mitigates memory overhead by training a compact model on uncertainty-selected instances, improving efficiency. Similarly, CSGD reduces hyperparameter sensitivity and computational cost through metaheuristic-driven optimization, enabling better handling of data complexity compared to KNN.
- SVM is known for high performance in small datasets but suffers from scalability, heavy memory usage due to kernel matrices, and lacks of streaming support. ASGD's iterative learning makes it highly scalable and efficient for large ETD datasets. CSGD eliminates manual kernel and hyperparameter tuning by leveraging CS to dynamically optimize SGD parameters.
- LR is interpretable and computationally light but assumes linearity and is limited in modeling complex or non-linear theft behavior. ASGD enhances LR-like models by adapting to complex patterns through focused learning on uncertain decision boundaries. By automatically tuning parameters like learning rate and regularization, CSGD enhances LR's convergence and recall, especially for rare fraudulent instances.
- TabNet leverages attention mechanisms for tabular data but is hyperparameter-sensitive, computationally expensive, and not designed to handle temporal or highly imbalanced datasets directly. ASGD offers a lightweight and adaptive alternative, prioritizing uncertain samples without the need for deep attention modules, which improves learning efficiency in class-imbalanced ETD. CSGD optimizes the model with fewer resources than TabNet by employing global search to tune hyperparameters without exhaustive trial-and-error, making it suitable for deployment in smart grid systems.
- CNN-GRU architectures can model spatial-temporal patterns but are computationally intensive and often act as black-box models with limited interpretability. They require large training data and are prone to overfitting, particularly under severe class imbalance. ASGD improves interpretability and robustness by focusing learning on uncertain instances and avoiding the smoothing artifacts of recurrent layers that overlook sharp fraud transitions. CSGD eliminates the need for manual tuning of sequence window sizes and learning rates by using CS to discover optimal configurations, resulting in stable training and better fraud detection under imbalanced settings.
- SGD is suitable for large-scale data but suffers from instability due to fixed hyperparameters and lacks mechanisms to handle imbalance or noise. ASGD improves stability and efficiency by actively selecting high-entropy samples that better guide the model's learning process. CSGD integrates CS to optimize SGD hyperparameters (learning rate, regularization), leading to more stable convergence and enhanced detection of minority class instances.

The proposed ASGD and CSGD models effectively address the shortcomings of existing ETD models, such as scalability, interpretability, hyperparameter sensitivity, and poor minority class detection. ASGD enhances learning efficiency through active sample selection to improve learning efficiency, while CSGD employs global metaheuristic optimization to ensure robust convergence and performance. Together, they form a lightweight, interpretable, and scalable solution well-suited for real-world ETD systems.

Table 7
Performance comparison of ASGD, and CSGD with baseline and state-of-the-art models for predicting electricity theft detection

Classifier	Accuracy	Precision	Recall	F1-score	ROC-AUC	PR-AUC	Execution Time (s)
KNN	0.76	0.72	0.79	0.72	0.74	0.84	0.84
SVM	0.73	0.78	0.72	0.76	0.78	0.80	213.22
LR	0.65	0.73	0.50	0.59	0.65	0.77	26.52
TabNet	0.62	0.57	0.98	0.72	0.88	0.79	683.64
CNN-GRU	0.50	0.50	0.98	0.66	0.50	0.71	799.47
SGD	0.60	0.69	0.37	0.48	0.66	0.61	459.03
Proposed CSGD	**0.81**	**0.77**	**0.88**	**0.82**	**0.86**	**0.90**	**244.43**
Proposed ASGD	**0.82**	**0.76**	**0.91**	**0.83**	**0.83**	**0.91**	**577.53**

4.3. Comparative Analysis of ASGD and CSGD with Benchmark Models

This study proposes a comparative analysis of proposed models with benchmark and baseline models. As shown in Table 7 and Figure 8, several classification models from existing literature for ET detection are compared and presented. ASGD and CSGD demonstrate superior performance when compared to benchmark techniques such as KNN, SVM, LR, TabNet, CNN-GRU, and baseline SGD. KNN shows moderate performance, with an F1-score of 0.72, precision of 0.72, recall of 0.79, and accuracy of 0.76. Although it identifies a significant number of fraudulent cases, it also yields a high number of FPs due to its distance-based nature and inability to generalize well in high-dimensional settings. SVM achieves an F1-score of 0.76 and accuracy of 0.73, but has the lowest recall of 0.72 among traditional classifiers. This indicates a reduced ability to detect fraudulent cases, making SVM less suitable for ETD, where recall is critical. Its batch-learning structure and sensitivity to kernel parameters limit its adaptability.

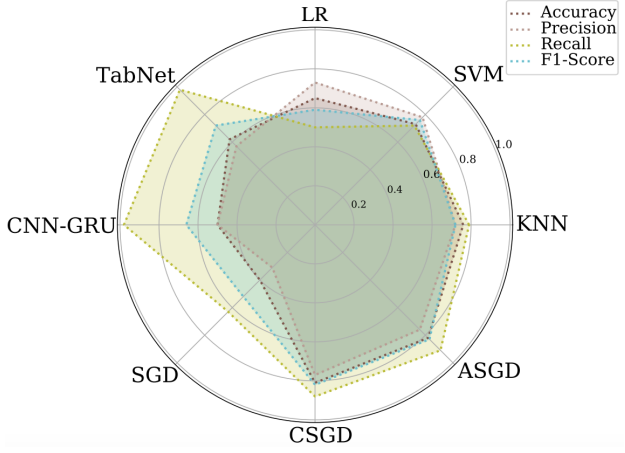


Figure 8: Performance comparison of state-of-the-art models with ASGD and CSGD for electricity theft detection

LR exhibits significant limitations in detecting complex fraud patterns, as reflected by its F1-score of 0.59, accuracy of 0.65, precision of 0.73, and very low recall of 0.50. Its assumption of linearity and sensitivity to imbalance hinders its effectiveness in ETD scenarios. TabNet, while using attention mechanisms for tabular data, also performs suboptimally. It records an accuracy of 0.62, an F1-score of 0.72, a precision of 0.57, and a very high recall of 0.98. The high recall suggests that it identifies most theft cases, but with many FPs, due to overfitting and complexity in hyperparameter tuning. The SGD baseline model performs poorly with the lowest F1-score of 0.48, accuracy of 0.60, precision of 0.69, and a recall of only 0.37. Its static nature, sensitivity to learning rate, and inability to handle imbalance or noise limit its performance in real-world ETD systems. Finally, the CNN-GRU model shows the weakest overall performance among all models, with an accuracy of 0.50, F1-score of 0.66, precision of 0.50, and recall of 0.98. Despite its excellent recall, the extremely low precision indicates a high number of FPs.

However, the proposed ASGD model outlined in Figure 3 and Algorithm 1, achieves a moderate recall of 0.91 and F1-score of 0.83, along with an accuracy of 0.82 and precision of 0.76. These results reflect its strength in identifying fraudulent cases while maintaining a well-balanced predictive performance. The use of entropy-based AL allows ASGD to focus on the most uncertain samples, enhancing learning efficiency and generalization. Its moderate precision reflects a deliberate bias toward recall, which is essential in ETD systems where missing a theft case is costlier than false alarms. Similarly, the proposed CSGD depicted in Figure 3 and Algorithm 1 follows closely, attaining an F1-score of 0.82, accuracy of 0.81, precision of 0.77, and recall of 0.88. The integration of CS optimization with SGD allows CSGD to dynamically tune hyperparameters, resulting in stable convergence and effective detection of minority (fraudulent) cases. The model's performance is both accurate and efficient, with robust generalization across data variations. The CNN-GRU model struggles to capture long-term temporal dependencies effectively. While CNNs are suitable for local feature extraction, they are not designed to handle sequential data patterns. GRUs, although

effective in modeling sequences, fail to capture irregular and abrupt theft behavior in electricity usage data. Additionally, CNN-GRU lacks advanced training enhancements such as AL and optimization techniques. This makes it prone to overfitting, especially in highly imbalanced datasets. Finally, the proposed models integrate both AL (ASGD) and bio-inspired optimization (CSGD), which significantly improve their learning focus, convergence, and fraud detection rates. ASGD targets high-uncertainty samples, optimizing model generalization, while CSGD uses global search to escape local minima and identify optimal hyperparameter configurations. These capabilities result in balanced, efficient, and scalable ETD models, surpassing the CNN-GRU and all benchmark classifiers across all key metrics.

The ROC-AUC comparison of several models for detecting ET is shown in Figure 9, illustrating the trade-off between TPR and FPR. The ASGD model achieves an ROC-AUC value of 0.83, as outlined in Algorithm 1 Line 20, which is significantly better than all existing models. This result demonstrates ASGD's strong ability to distinguish between fraudulent and non-fraudulent cases with high confidence. The main reason for ASGD's remarkable performance is its AL-based approach, which efficiently refines decision boundaries and prioritizes uncertain instances. This approach ensures robust fraud detection by allowing the model to maximize recall while maintaining high precision. SGD and CS optimization are combined to provide the advantage of dynamic hyperparameter tuning. By enhancing the model's generalization and decreasing overfitting, this procedure improves the detection of fraudulent activity. Although it optimizes learning effectively, its comparatively higher ROC-AUC of 0.86, when compared to ASGD, indicates that it is better suited to handle highly uncertain cases. KNN has an ROC-AUC of 0.74. However, KNN is a lazy learner that results in slow inference and high computational costs, especially as the dataset increases.

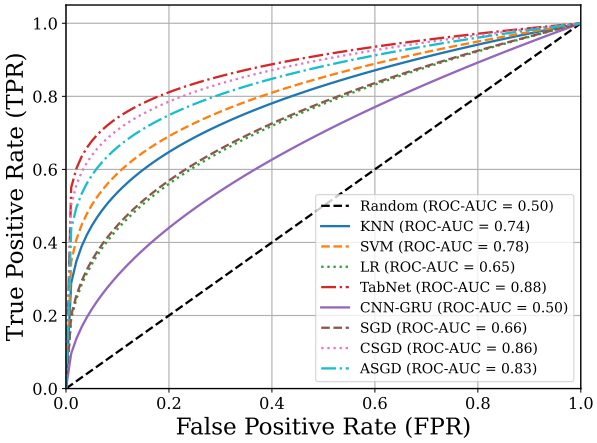


Figure 9: ROC-AUC comparison of proposed ASGD and CSGD with state-of-the-art models for electricity theft detection

The reason for this problem is that KNN depends on distance metrics, which are susceptible to changes in feature distributions. The performance of LR and SVM is significantly less fortunate, with ROC-AUCs of around 0.65 and 0.78. LR is sensitive to multicollinearity, in that correlated features have the potential to skew the model's coefficient estimates, making interpretation challenging. SVM, on the other hand, enforces rigid decision boundaries, which results in low recall, making it less effective in contexts like ETD where identifying fraudulent users is critical. Similar to this, LR performs less well in classification because it relies on linear decision functions, which hinders it from identifying complex fraud patterns. SGD achieves an ROC-AUC of 0.66, mostly due to its simple gradient updates cannot manage situations with high levels of uncertainty. The baseline SGD model suffers the most, producing weak classification boundaries. TabNet demonstrates competitive performance with an ROC-AUC of 0.88, slightly higher than ASGD. However, its precision is lower, which indicates that despite its strong ability to distinguish classes, it suffers from higher FPRs. The results suggest a tendency toward overfitting or sensitivity to imbalance, which ASGD and CSGD handle more effectively through AL and optimization. Finally, the DL hybrid CNN-GRU model exhibits the weakest performance, registering a ROC-AUC of only 0.50. This is equivalent to random guessing and highlights the model's failure to distinguish fraudulent from honest cases effectively. Although CNN-GRU achieves high recall, its very low precision and poor ROC-AUC emphasize its over-reliance on positive class predictions without learning adequate decision boundaries. Overall, the ROC-AUC analysis highlights the superior classification reliability of the proposed ASGD and CSGD models. Their ability to balance FPs and TPs, along with robust ROC-AUC values, highlights their effectiveness in real-world electricity theft detection scenarios.

Additionally, the PR-AUC metric provides an insightful evaluation of model performance, especially in imbalanced datasets. In our experiments, the proposed models, ASGD and CSGD, demonstrate superior performance compared to the baseline models, as shown in Table 7 and Figure 10. The proposed CSGD achieves a PR-AUC of 0.90, which is notably higher than other baseline models. This improvement is due to the CSGD model's Algorithm 1 ability to better balance the model's exploration and exploitation in the optimization process, thus resulting in more accurate predictions, especially in scenarios where the data is imbalanced. Similarly, the proposed ASGD achieves a PR-AUC of 0.91, surpassing other models by actively selecting the most uncertain data points for learning, which further enhances its ability to detect ET. In contrast, KNN, SVM, and LR models, with PR-AUC values of 0.84, 0.80, and 0.77, respectively, exhibit less effectiveness in capturing the intricacies of imbalanced data, as they lack mechanisms to adapt to the challenging nature of fraudulent consumption detection. These models struggle to improve precision without sacrificing recall, resulting in suboptimal performance in detecting fraud. Similarly, TabNet and CNN-GRU models, while achieving reasonable PR-AUC scores of 0.79 and

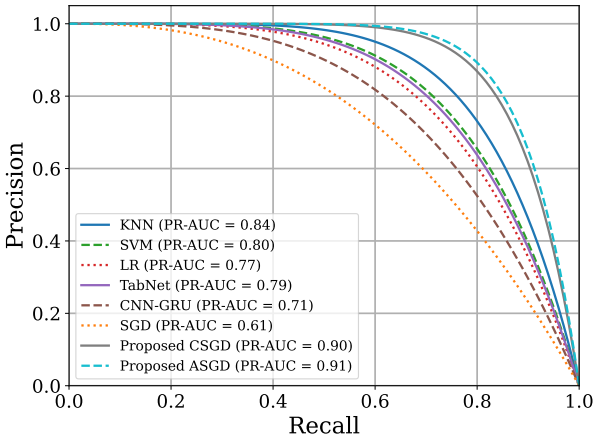


Figure 10: PR-AUC comparison of proposed ASGD and CSGD with state-of-the-art models for electricity theft detection

0.71, still face issues with scalability, making them less suitable for advanced ETD applications. These models, especially CNN-GRU, benefit from DL features but require substantial computational resources for training. The significantly higher PR-AUC scores of ASGD and CSGD highlight their superior ability to efficiently learn from complex non-linear data, making them ideal for the task of ETD, where the majority class (Honest) vastly outnumbers the minority class (Fraudulent).

The proposed ASGD and CSGD model execution times are compared to existing models shown in Figure 11, which clarifies the level of computational effectiveness of each approach. Among the baseline models, KNN demonstrates the fastest execution time of 0.84 seconds; however, its relatively moderate accuracy of 0.76 and F1-score of 0.72 limit its utility in high-stakes applications. KNN also scales poorly during inference in larger datasets due to its lazy learning nature. LR is computationally efficient with an execution time of 26.52 seconds but suffers from a low recall of 0.50 and accuracy of 0.65, largely due to its linear decision boundary, making it ill-suited for capturing complex patterns in theft behavior. SVM, while offering a decent balance in metrics with ROC-AUC of 0.78, has a significantly higher execution time of 213.22 seconds, making it computationally expensive as the data grows. TabNet and CNN-GRU demonstrate very high execution times (683.64 seconds and 799.47 seconds, respectively), attributed to their DL architectures. While TabNet exhibits a strong recall of 0.98, its precision and accuracy are weak, showing instability. Similarly, CNN-GRU attains a high recall of 0.98, but at the cost of poor accuracy of 0.50, F1-score of 0.66, and significant computational demand, rendering it inefficient for deployment in time-sensitive environments. SGD, a basic optimizer, shows a moderate execution time of 459.03 seconds but underperforms in recall with 0.37 and F1-score with 0.48, often converging prematurely due to its stochastic nature and lack of adaptability in imbalanced scenarios. Finally, the proposed CSGD and ASGD not only outperform all benchmarks in classification metrics but also offer a reasonable trade-off in execution time. CSGD achieves an accuracy of 0.81 and an F1-score of 0.82 in just 244.43 seconds. The integration of the CSGD

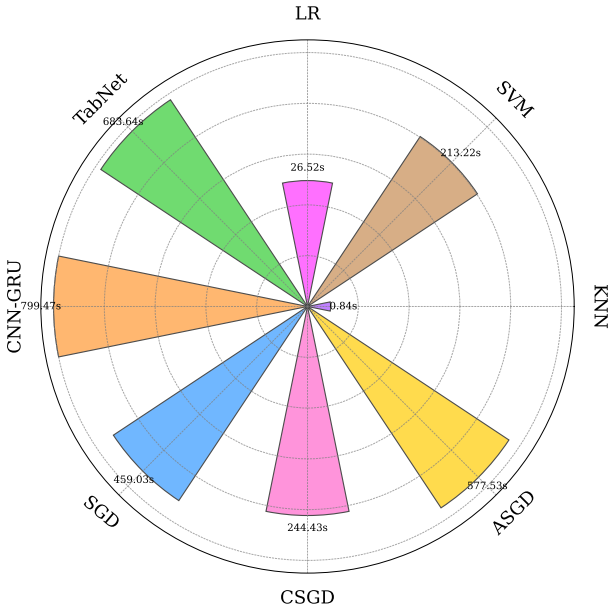


Figure 11: Execution time comparison of proposed ASGD and CSGD with state-of-the-art models for electricity theft detection

algorithm enables efficient global optimization, helping the model escape local minima and enhancing generalization. ASGD slightly outperforms CSGD with an accuracy of 0.82 and an impressive recall of 0.91, due to its entropy-driven sampling that focuses on the most informative samples. Although ASGD requires more time of 577.53 seconds, its efficiency in training with fewer, high-impact data points justifies the additional computational cost. Conclusively, while benchmark models like KNN, LR, SVM, TabNet, and CNN-GRU suffer from either poor performance or excessive computation, CSGD and ASGD offer a compelling balance of accuracy, robustness, and execution efficiency, making them highly suitable for intelligent ETD systems.

The architecture of ASGD is mentioned in Table 8, explaining its learning process and computational structure systematically. The representation of the input characteristics: each data sample consists of two features, as indicated by the $n \times 2$ format (Line 1). A random initialization approach is used for θ and bias b. Given that $\theta \in \mathbb{R}^{2\times 1}$ and $b \in \mathbb{R}$, the ASGD begins with non-deterministic parameters for efficient learning (Lines 2–3). The ASGD is suitable for use in binary classification because the activation function explains what is necessary to transfer outputs to a probability range of 0 to 1 (Line 5). To provide a flexible modification of ASGD parameters according to calculated gradients, the SGD optimization method is employed (Line 6). Binary Cross-Entropy (BCE), a commonly used objective function in classification problems, is the loss function that is employed in (Line 7). A learning rate of 0.01 is adopted (Line

Table 8
Architecture of the proposed ASGD and CSGD models

ASGD Architecture	CSGD Architecture
1. Input Features (shape = $n \times 2$)	1. Input Layer
2. Weights Initialization: $\theta \in \mathbb{R}^{2\times1}$ (random initialization)	2. Input features: $n\_features$
3. Bias Initialization: $b \in \mathbb{R}$ (random initialization)	3. Number of nests: 20
4. Training Process	4. Number of generations: 10
5. Sigmoid Function: $\sigma(z) = \frac{1}{1+e^{-z}}$	5. Abandon probability (p_a): 0.25
6. Stochastic Gradient Descent (SGD)	6. Objective function: Log Loss $= -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(p_i) + (1-y_i)\log(1-p_i)]$
7. Loss Function: Binary Cross-Entropy	7. Optimization for: Weights (θ) and Bias (b)
8. Learning Rate: 0.01	8. Sigmoid Activation
9. Iterations: 1000 per active learning iteration	9. Function: $\sigma(z) = \frac{1}{1+e^{-z}}$
10. Active Learning Process	10. Stochastic Gradient Descent
11. Uncertainty Sampling: Based on margin $\lvert 0.5 - \hat{y}\rvert$	11. Learning rate: 0.01
12. Batch Size for Sampling: 32 most uncertain instances	12. Number of epochs: $n\_iterations = 3$
13. Number of Active Learning Iterations: 100	13. Weight update rule: $\theta = \theta - \alpha \cdot d\theta$
	14. Bias update rule: $b = b - \alpha \cdot db$
	15. Prediction Function
	16. Probability: $p(X) = \sigma(X \cdot \theta + b)$
	17. Binary Prediction: $p(X) \geq 0.5 \rightarrow 1$, else 0

8) to ensure stable convergence and prevent weight update oscillations. Each AL cycle has 1000 SGD iterations (Line 9), ensuring that there are sufficient training epochs for model improvement. ASGD includes an uncertainty-based AL procedure that prioritizes instances close to the decision boundary. The most ambiguous examples are identified for annotation and retraining using the metric (Line 11). To refine the model with high-impact samples, batches of the 10 most uncertain instances are selected during each iteration (Line 12). In particular, (line 13) mentions the number of AL iterations, which is 100. In other words, the ASGD model goes through 100 cycles of AL, during which it iteratively annotates 400 samples on each cycle on its decision boundary, updates its weights, and chooses the most uncertain samples.

In contrast, the CSGD architecture, detailed in Table 8, provides a comprehensive explanation of its computational structure and learning process. The input layer, the initial layer of the CSGD, receives input (Lines 1-2). In order to ensure robust exploration and exploitation of the solution space, the model is optimized using CS with the number of nests set to $N = 8$ and the number of generations defined as $T = 10$ (Line 5). The frequency at which the worst solutions are discarded and replaced with new ones is determined by the abandon probability $p_a = 0.25$. (Line 6). The objective function used in the optimization process is log-loss, a commonly used metric in classification tasks to quantify prediction errors and guide weight adjustments. To ensure an adaptable learning approach (Line 7), the optimization method focuses on fine-tuning the weights (θ) and bias (b) (Lines 8–9). The *sigmoid* activation function converts raw scores into probability values between 0 and 1 (Lines 10–11).. For weight updates, the SGD approach is used, providing consistent convergence with a predetermined learning rate of $\eta = 0.01$ (Line 12). For efficient learning, the number of epochs is set to $n_{\text{iterations}} = 2$, enabling several runs through the dataset. Lines 13–14 provide the weight update rule and the bias update rule, where α represents the learning rate. The probability is calculated by the prediction function (Lines 15–16). The binary classification decision rule is applied (Line 17), where instances are classified as 1 (positive class), while the rest are classified as 0 (negative class).

Similarly, the hyperparameters in Table 9 are used to build the ASGD model such that ET is detected as efficiently as feasible. By avoiding abrupt changes that could lead to oscillation or overshooting the ideal solution, a learning rate of 0.01 (Line 8) guarantees steady and gradual learning. The number of SGD iterations 1000 (Line 9) specifies how many iterations the model goes through to update its weights using SGD, striking a balance between computational efficiency and adequate training. The model's decision boundaries are improved by ASGD's iterative selection of high-uncertainty instances through the use of 100 AL iterations. The testing

Table 9
Hyperparameters utilized in ASGD and CSGD Models

ASGD Model		CSGD Model	
Hyperparameter	**Value**	**Hyperparameter**	**Value**
Penalty	l2	Alpha (α)	$1e^{-6}$ to $1e^{-1}$
Max Iterations	1000	Eta0 (η_0)	0.0001 to 0.2
Tolerance (tol)	10^{-3}	Max Iterations	1000
Random State	42	Tolerance (tol)	10^{-3}
Batch Size (AL)	32	Random State	42
Number of Iterations (AL)	100	Nest	8
Annoated Samples per Iteration	400	n_iterations (Cuckoo)	3

Table 10
Performance comparison proposed CSGD and state-of-the-art optimization techniques

Classifier	Accuracy	Precision	Recall	F1-score	ROC-AUC	PR-AUC	Execution Time (s)
PSO-SGD	0.75	0.75	0.77	0.76	0.81	0.77	1598.59
SBO-SGD	0.76	0.72	0.84	0.78	0.82	0.70	833.05
Proposed CSGD	**0.81**	**0.77**	**0.88**	**0.82**	**0.86**	**0.90**	**244.43**

process utilizes 20% of the dataset to accurately assess the model's capacity for generalization. By regulating the generation of random numbers, *Random State* is set to 42, which guarantees reproducibility by producing consistent data splits and parameter initialization throughout simulations.

On the other hand, Table 9 provides specifics on the hyperparameter setup used in the CSGD model for ETD. The learning rate parameter α is defined within the range $(10^{-6}, 10^{-1})$, allowing for adaptive tuning within a broad scale to optimize model convergence. Furthermore, to provide steady updates during optimization, the initial learning rate η_0 is limited between 0.0001 and 0.2. The maximum number of iterations the model can undergo is *max_iter* = 1000, which ensures that there are sufficient training cycles for convergence. Additionally, the CS optimization component incorporates *n_iterations* = 3, which indicates three iterations for search process optimization. In order for the CS mechanism to efficiently explore the solution space, the number of nests used is indicated by the *n_nests* parameter, which is set to 15. All of these hyperparameters work together to enhance the CSGD model's robustness and predictive performance, particularly in accurately identifying ET instances with high stability.

4.4. Comparison of CSGD with Existing Optimization Techniques

In this section, we present a comparative analysis of the two well-established optimization techniques: PSO [16] and SBO [21] optimization. The comparison covers various performance metrics, including accuracy, precision, recall, F1-score, ROC-AUC, PR-AUC, and execution time.

As shown in Table 10 and Figure 12, the proposed CSGD model presented in Algorithm 2 and Figure 3, outperforms the existing optimization techniques, PSO-SGD and SBO-SGD, in several key metrics. Specifically, the accuracy of CSGD is 0.81, surpassing both PSO-SGD with 0.75 and SBO-SGD with 0.76. This demonstrates that the proposed CSGD model not only provides a higher level of accuracy but also yields more consistent and reliable results across multiple performance metrics. The enhanced performance of the proposed CSGD can be attributed to the inherent advantages of the CS algorithm, which has been proven to be effective in exploring complex search spaces with better convergence properties than PSO and SBO. Moreover, the precision, recall, and F1-score of the proposed CSGD are also superior compared to the existing techniques, as shown in Figure 12c. With a recall of 0.88, CSGD outperforms PSO-SGD with 0.77 and SBO-SGD with 0.84. This indicates that the proposed CSGD not only has higher accuracy but also exhibits better sensitivity in identifying TP instances, particularly in imbalanced datasets. Similarly, the F1-score for CSGD with 0.82 is higher than that of PSO-SGD with 0.76 and SBO-SGD with 0.78, signifying a better balance between precision and recall. Additionally, the ROC-AUC of the proposed CSGD 0.86 is better than that of both PSO-SGD with 0.81 and SBO-SGD with 0.82 given in Figure 12a, further confirming the robust performance of the CS-based optimization in distinguishing between positive and negative classes in classification tasks. In terms of PR-AUC, as shown in Table 10 and Figure 12b, the proposed CSGD achieves a higher PR-AUC of 0.90 compared to PSO-SGD with

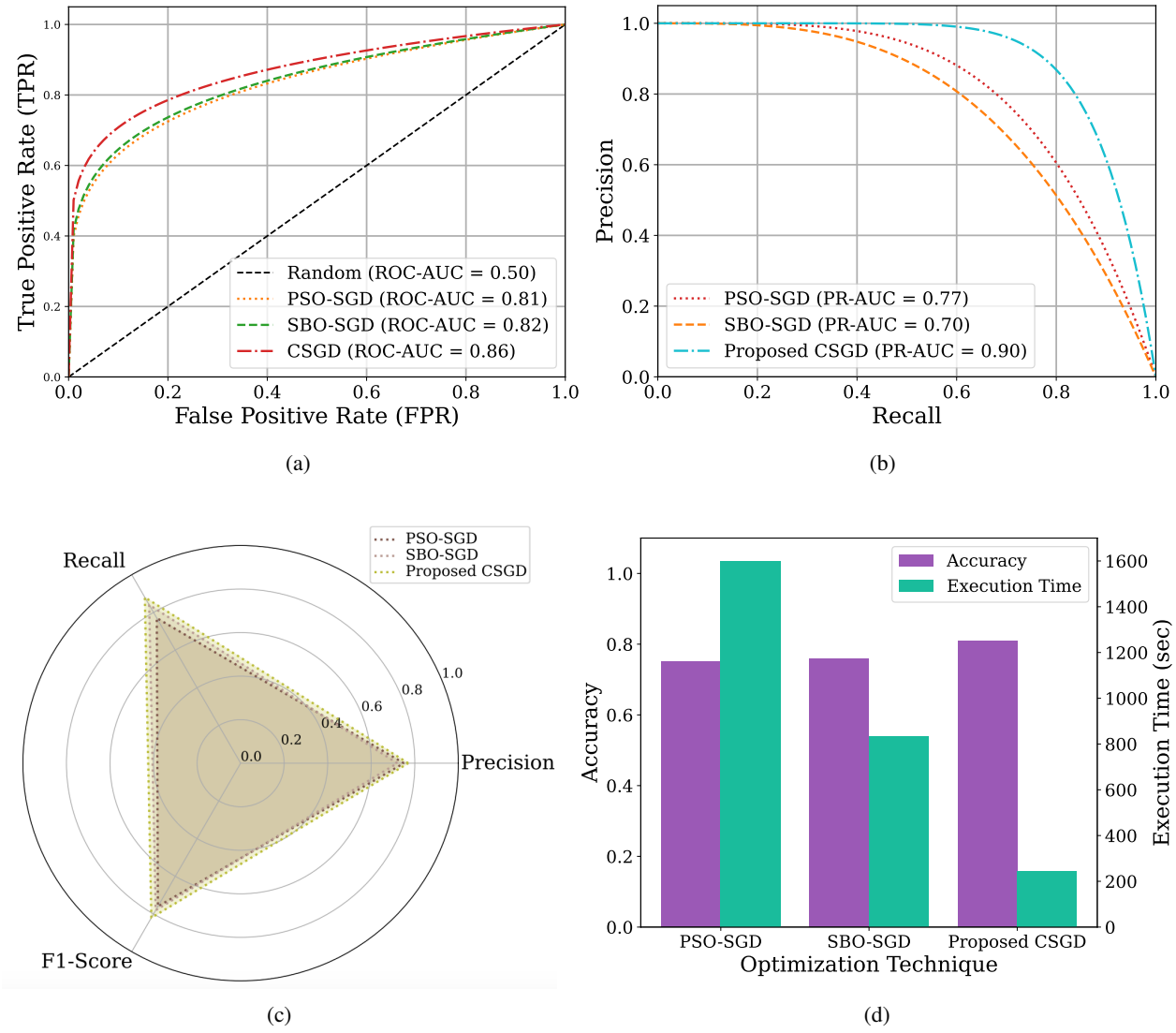


Figure 12: Performance comparison of state-of-the-art optimization techniques (PSO-SGD, SBO-SGD) with the poroposed CSGD across different metrics: (a) ROC-AUC, (b) PR-AUC, (c) Radar plot (Precision, Recall, F1-score), and (d) Accuracy and Execution Time.

0.77 and SBO-SGD with 0.70. This improvement in PR-AUC indicates that the proposed CSGD is more effective at distinguishing between positive and negative samples in imbalanced datasets, where traditional accuracy may not provide a clear picture of model performance. The higher PR-AUC of CSGD further supports its advantage over the existing optimization techniques in scenarios with a skewed class distribution. One of the key advantages of the proposed CSGD is its improved performance in both accuracy and execution time. As shown in Table 10 and Figure 12d, the accuracy of CSGD is 0.81, surpassing PSO-SGD with 0.75 and SBO-SGD with 0.76. Furthermore, the execution time for CSGD is 244.43 seconds, which is significantly lower than PSO-SGD with 1598.59 seconds and SBO-SGD with 833.05 seconds. The faster convergence of CSGD can be attributed to the efficient exploration–exploitation trade-off of the CS algorithm, which accelerates convergence by narrowing down the search space more effectively and minimizes the need for extensive iterations. In terms of accuracy, CSGD achieves superior performance with an accuracy of 0.81, indicating a higher level of precision in detecting electricity theft compared to both PSO-SGD and SBO-SGD. This is particularly advantageous when working with imbalanced datasets where correctly identifying fraudulent behavior is crucial. On the other hand, the reduced execution time for CSGD demonstrates its computational efficiency. The CS algorithm's faster convergence is due to its ability to avoid getting stuck in local minima, a common challenge faced by PSO. PSO's stochastic particle movements often result in slower convergence, and its parameter sensitivity complicates fine-tuning for a variety of problems. SBO, while providing more structure than PSO, still struggles with balancing exploration and exploitation, often resulting in overfitting and reduced generalization. Additionally, the

Table 11
Performance comparison of proposed ASGD and state-of-the-art active learning techniques

Classifier	Accuracy	Precision	Recall	F1-score	ROC-AUC	PR-AUC	Execution Time (s)
MCAL-SGD	0.74	0.72	0.78	0.75	0.75	0.77	174.41
CSAL-SGD	0.78	0.84	0.73	0.78	0.86	0.78	18.57
Proposed ASGD	**0.82**	**0.76**	**0.91**	**0.83**	**0.83**	**0.91**	**577.53**

increased computational complexity of SBO with larger problem sizes makes it less practical for large-scale tasks. In contrast, the proposed CSGD algorithm demonstrates superior computational efficiency, offering both enhanced accuracy and faster convergence. This makes it a highly effective solution for real-world optimization problems where computational resources and time are critical constraints.

Additionally, the CS algorithm used in the proposed CSGD offers several advantages over PSO and SBO. First, it has an inherent ability to avoid local minima due to the use of Levy flights for global search, enabling it to explore the search space more effectively. The CS's adaptive nature also helps in balancing exploration and exploitation better than PSO and SBO, leading to faster convergence to optimal solutions. Moreover, the proposed CSGD algorithm is computationally efficient and scales well with large datasets, as seen from the reduced execution time compared to PSO and SBO. This makes CSGD a highly practical choice for real-world optimization problems where time and computational resources are critical constraints.

4.5. Comparison of ASGD with Existing Active Learning Techniques

This section compares the performance of the proposed ASGD (Entropy-based AL SGD) depicted in Algorithm 1 and Figure 3 with two widely used AL techniques: Monte Carlo AL SGD (MCAL-SGD) [14] and Core-Set AL SGD (CSAL-SGD) [6]. The evaluation is based on several key metrics: accuracy, precision, recall, F1-score, ROC-AUC, PRAUC, and execution time.

Figure 13 provides a comprehensive visual comparison of these methods, while Table 11 summarizes the quantitative results. We begin with a comparison of the ROC-AUC, which is a crucial metric for evaluating the ability of the models to distinguish between the positive and negative classes. As shown in Figure 13a, the proposed ASGD achieves the highest ROC-AUC of 0.83, outperforming MCAL-SGD with 0.75 and CSAL-SGD with 0.86 by a significant margin. While CSAL-SGD achieves a high ROC-AUC score, the proposed ASGD excels in providing a better trade-off between sensitivity and specificity. This performance can be attributed to the entropy-based selection strategy of ASGD on ambiguous samples. ASGD enhances the model's ability to differentiate between classes, resulting in improved performance.

Moving to PR-AUC in Figure 13b, the proposed ASGD achieves a PR-AUC of 0.91, which is higher than both MCAL-SGD with 0.77 and CSAL-SGD with 0.78. This demonstrates that ASGD not only maintains a satisfactory balance between precision and recall but also improves the model's ability to identify rare positive instances, which is critical for applications like fraud detection. The PR-AUC score highlights ASGD's superiority in prioritizing informative instances during AL training, thereby significantly enhancing its ability to recognize fraud while avoiding FPs.

Next, in Figure 13c, we present a comparison of precision, recall, and F1-score across all three techniques. ASGD stands out with the highest values for recall with 0.91 and F1-score with 0.83, indicating that the model is very effective in detecting fraudulent instances (TPs) without sacrificing precision. On the other hand, CSAL-SGD shows the highest precision with 0.84, but it is less effective in recall with 0.73, which means it struggles to capture all potential positive instances. The proposed ASGD strikes a more favorable balance, ensuring that while precision is slightly lower than CSAL-SGD, the model's recall is far superior, making it more robust in handling time series data.

Finally, accuracy and execution time in Figure 13d show a clear trade-off. While ASGD achieves the highest accuracy of 0.82, it requires the longest execution time of 577.53 seconds compared to MCAL-SGD of 174.41 seconds and CSAL-SGD of 18.57 seconds. This higher computational cost is a result of the more complex entropy-based AL approach, which selects instances based on model uncertainty and continuously adapts the model as it learns. Despite the longer runtime, the substantial performance improvements in accuracy, precision, recall, and F1-score make ASGD the most effective technique for tasks where both recall and precision are critical, and where computational resources are not a primary limitation. Conclusively, the proposed ASGD algorithm not only outperforms existing techniques in terms of recall and F1-score but also excels in precision and ROC-AUC performance. However, it comes at the cost of higher computational resources, which is a consideration for large-scale applications. Nevertheless, its ability to balance predictive performance with high recall it an ideal choice for challenging tasks, particularly in fraud detection and other applications where recall is crucial for capturing rare events.

4.6. Maximizing ASGD Model Performance with Smart Annotations

The Table 12 shows a model's performance metrics based on the various annotation sizes, illustrating the effects of variations in the number of annotated instances on accuracy, precision, recall, F1-score, and ROC-AUC. Annotation

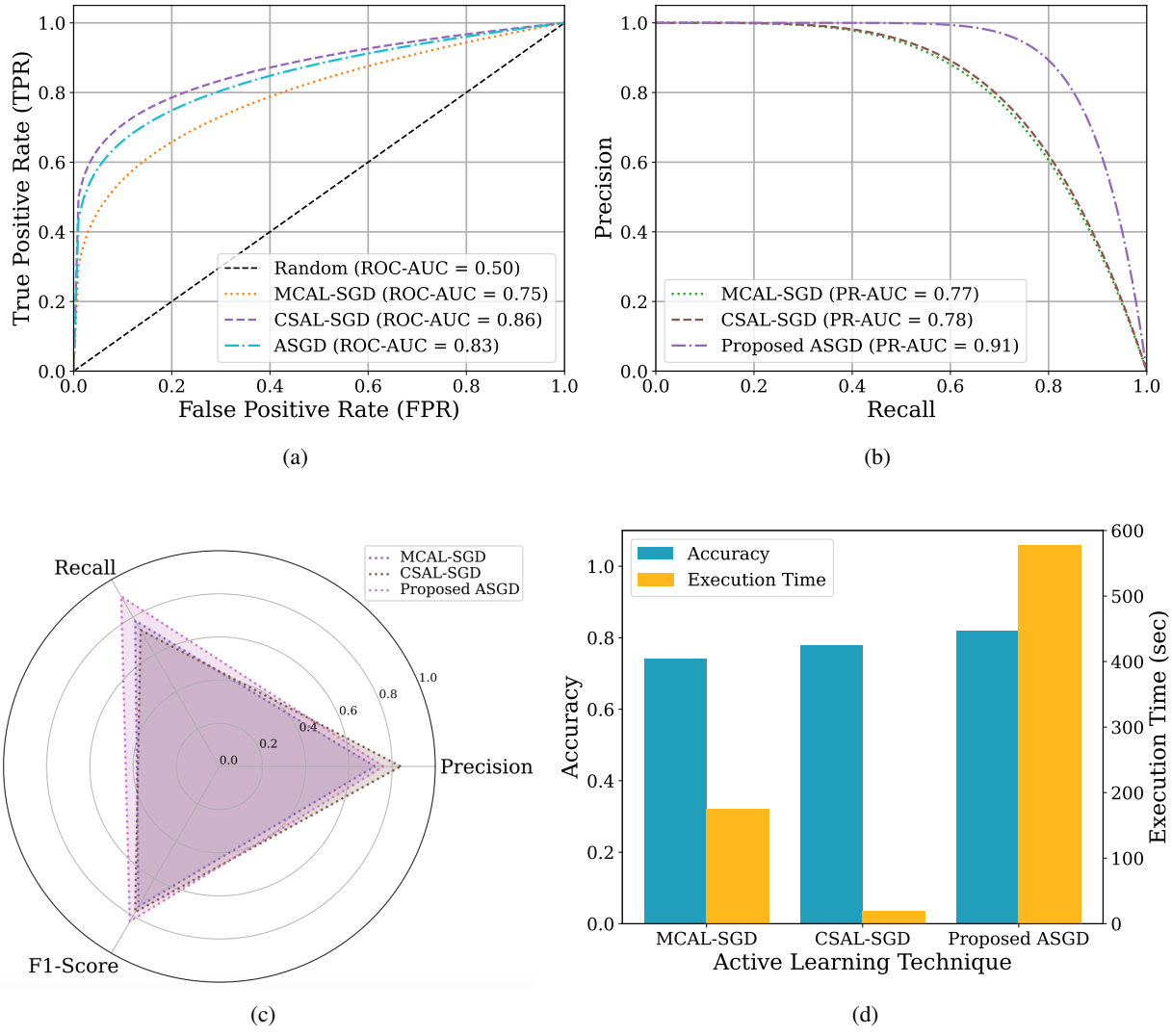


Figure 13: Performance comparison of state-of-the-art AL techniques (MCAL-SGD, CSAL-SGD) with the proposed ASGD across different metrics: (a) ROC-AUC, (b) PR-AUC, (c) Radar plot (Precision, Recall, F1-score), and (d) Accuracy and Execution Time.

Table 12
Performance metrics for different annotation sizes

Metrics	Annotations				
	2000	**3000**	6000	8000	Full Data
Accuracy	0.66	**0.82**	0.63	0.66	0.78
Precision	0.62	**0.76**	0.73	0.70	0.71
Recall	0.83	**0.91**	0.43	0.56	0.74
F1-score	0.71	**0.83**	0.54	0.62	0.72
ROC-AUC	0.67	**0.83**	0.66	0.68	0.73

size and performance have a non-linear connection; the results indicates that the optimum overall performance is achieved with 3000 annotations, after which performance begins to degrade with further increases in annotation size. Several factors contribute to this discrepancy, including overfitting, label quality, data variety, and model learning capacity. The ASGD model's accuracy of 0.66, when trained on 2000 annotations, indicates that while the data provides some significant patterns, it is still insufficient for reliable generalization. The ASGD model is generating a considerable amount of fraudulent cases, as evidenced by the comparatively poor precision of 0.62. The recall of 0.83 indicates that the model is very sensitive to identifying positive cases, even at the expense of accuracy. Despite this, the F1-score of 0.71 highlights a lack of balance between precision and recall, showing that the model's predictions are not yet well-calibrated for producing consistent results across both dimensions. The ASGD model's current ability to discriminate between positive and negative classes is moderate, according to the ROC-AUC of 0.67.

The accuracy rises to 0.82, indicating that the model has effectively learned generalizable patterns and is operating at peak performance across all measures at 3000 annotations. With a precision of 0.76, fraudulent cases have decreased, resulting in forecasts that are more trustworthy. The model's recall rises to 0.91, demonstrating its ability to accurately identify a high percentage of positive cases. Consequently, the F1-score reaches 0.83, indicating an ideal balance between recall and precision. When it comes to class differentiation, the ROC-AUC of 0.83 shows the best performance. This implies that the dataset has a balanced representation of classes at 3000 annotations, allowing the model to learn strong decision boundaries free from undue bias or noise.

However, performance starts to suffer as the annotation size grows more than 3000. At 6000 annotations, the accuracy drops to 0.63, significantly lower than at 3000. This decrease implies that the model is unable to maintain high performance because of label discrepancies or redundant information introduced by the additional data. While the precision remains relatively high at 0.73, it is still lower than the precision achieved with 3000 annotations, indicating a slight increase in FPs. It is really concerning that the model is currently missing numerous useful instances, as seen by the sharp drop in recall to 0.43. A significantly lower F1-score of 0.54 results from this imbalance, indicating that the trade-off between precision and recall is no longer reliable at this scale.

The accuracy shows a modest improvement to 0.66 after 8000 annotations, although it is still far below the 3000 peak. The ASGD model is identifying more positive occurrences than at 6000, but still significantly fewer than at 3000, as evidenced by the recall improving marginally to 0.56. As a result, the F1-score increases marginally to 0.62, although it is still far below its ideal level. Class separability shows some recovery, as indicated by the ROC–AUC score of 0.68, but this value still falls short of the previous peak achieved at 3000 annotations. These variations suggest that the ASGD model has trouble overfitting to common patterns while underperforming at generalizing to less frequent situations when additional data is added. Lastly, the model, once trained on the entire dataset, falls short of the peak at 3000 annotations. Although it is still below the best-performing scenario, the accuracy reaches 0.78, which is higher than at 6000 and 8000 annotations. Though not ideal, the precision stays at 0.71, indicating a reasonably balanced level of fraudulent cases. The recall increases to 0.74, indicating that the model is capturing more positive cases than in previous larger-annotation scenarios, though still fewer than at 3000. The trade-off between recall and precision is better with an F1-score of 0.72. Although it still falls short of the peak at 3000 annotations, the ROC-AUC score of 0.73 indicates improved class separation.

4.7. 10-Fold Cross-Validation of Proposed ASGD and CSGD Models

10-FCV is a statistical method that divides a dataset into 10 equal-sized subsets or folds to assess the performance of an ML model. In each iteration, the model is tested on one fold after being trained on nine folds [38]. In equation 20, D is the dataset and N represents the total number of instances $\frac{N}{10}$.

$$D_{\text{train}} = D \setminus D_i, \quad D_{\text{test}} = D_i \tag{20}$$

The model is trained on D_{train} and evaluated on D_{test}, producing a metric M_i, such as accuracy, precision, recall, F1-score, ROC-AUC, and PR-AUC. The final metric is computed as the average across all folds using:

$$M_{\text{final}} = \frac{1}{10} \sum_{i=1}^{10} M_i \tag{21}$$

where M_{final} represents the final averaged metric, M_i denotes the metric value at the i-th iteration, $\sum_{i=1}^{10} M_i$ is the total sum of the metric values over 10 iterations, and $\frac{1}{10}$ is the normalization factor for computing the mean.
This method is beneficial as it offers a reliable approximation of the model's capacity to generalize to new data. Testing on various subsets lowers the possibility of overfitting, ensures effective dataset use, and strikes a balance between bias and variance. Furthermore, averaging results across folds reduces the influence of random variations in the data splits and provides trustworthy evaluation metrics. The flow diagram of 10-FCV is illustrated in Figure 14.

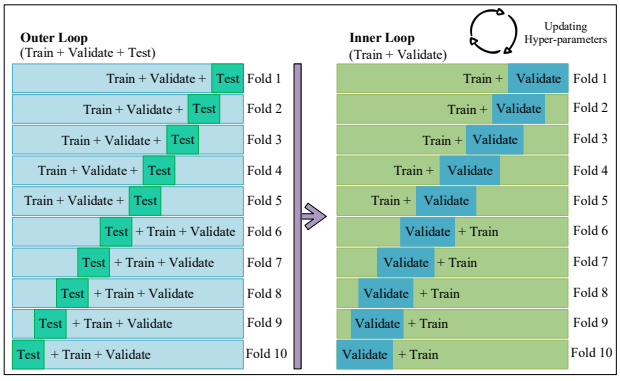


Figure 14: The working of 10-fold cross-validation technique

Several metrics are used in Table 13 to compare the performance of the proposed models. The first bold value for each metric indicates the outcome of the proposed models, the last bold value shows an average of 10 folds, and all other middle values indicate each fold's value. For 10-FCV, the dataset is divided into ten folds, and the proposed models are trained and tested on different folds. This method assesses the ASGD and CSGD models' capacity for generalization and lowers the possibility of overfitting. It provides a consistent overall score for the models' performance by calculating the mean of the individual fold

Table 13
Results of proposed ASGD and CSGD with 10-fold cross-validation

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC	PR-AUC
SGD	0.60	0.69	0.37	0.48	0.66	61
ASGD with 10-FCV	**0.82**, 0.81, 0.67, 0.81, 0.77, 0.60, 0.72, 0.79, 0.67, 0.71, 0.78, **0.80**	**0.76**, 0.74, 0.66, 0.81, 0.85, 0.63, 0.65, 0.68, 0.64, 0.64, 0.69, **0.76**	**0.91**, 0.93, 0.68, 0.81, 0.66, 0.49, 0.75, 0.96, 0.78, 0.90, 0.88, **0.90**	**0.83**, 0.83, 0.67, 0.81, 0.74, 0.79, 0.70, 0.67, 0.77, 0.75, 0.80, **0.81**	**0.83**, 0.82, 0.77, 0.85, 0.85, 0.75, 0.77, 0.68, 0.72, 0.70, 0.69, **0.79**	**0.90**, 0.90, 0.87, 0.92, 0.88, 0.91, 0.89, 0.93, 0.90, 0.87, 0.91, **0.90**
CSGD with 10-FCV	**0.82**, 0.80, 0.74, 0.70, 0.81, 0.81, 0.80, 0.73, 0.78, 0.79, 0.80, **0.80**	**0.77**, 0.81, 0.81, 0.76, 0.80, 0.80, 0.81, 0.79, 0.78, 0.80, 0.77, **0.77**	**0.88**, 0.78, 0.75, 0.88, 0.82, 0.83, 0.77, 0.64, 0.70, 0.83, 0.79, **0.86**	**0.82**, 0.79, 0.72, 0.66, 0.81, 0.81, 0.79, 0.71, 0.79, 0.83, 0.77, **0.81**	**0.86**, 0.86, 0.83, 0.79, 0.86, 0.86, 0.86, 0.82, 0.85, 0.81, 0.79, **0.86**	**0.88**, 0.88, 0.89, 0.90, 0.88, 0.91, 0.89, 0.90, 0.91, 0.89, 0.90, **0.89**

results. Table 13 shows the performance of the Proposed ASGD and CSGD models with 10-FCV against the baseline SGD model. Both ASGD and CSGD outperform SGD in all key metrics: accuracy, precision, recall, F1-score, ROC-AUC, and PR-AUC. ASGD and CSGD achieve an accuracy of 0.82, a recall of 0.91 and 0.88, and ROC-AUC values of 0.83 and 0.86, respectively, compared to SGD's 0.60, 0.37, and 0.66. While ASGD and CSGD require higher execution times (577.53 and 244.43 seconds, respectively, as given in Table 7), their superior performance justifies the additional computational cost, particularly in tasks requiring high precision, recall, and class separation. The adoption of 10-FCV in our study was predetermined by a number of reasons that coincided with the topics of this research and the nature of the SGCC dataset. The value of 10-FCV is good in terms of bias and variance, which is necessary to guarantee the generalization of the model, especially in cases of high-dimensional datasets. The proposed approach provides stable performance estimates with minimal overfitting, which is of imperative importance when conducting ETD. Furthermore, 10-FCV represents the standardized assessment technique that can be compared to the other research directly. Even though leave-one-out cross-validation could be considered for its exhaustive evaluation by training on nearly all data points and testing on a single one, it is computationally expensive, especially for large datasets. It also shows a greater variance in performance estimation because it is so sensitive to individual data and hence not as useful in the necessities of our study. Stateside, stratified K-FCV guarantees that each stratum has the same percentage of classes, which is an advantage when the datasets are complex. Nevertheless, although it removes some bias compared to standard K-FCV, 10-FCV seems to be more comprehensive with more folds, consequently being more reliable and stable in providing model performance estimates.

4.8. Statistical Validation of ASGD and CSGD Performance Enhancements via Paired t-Test Analysis

The t-statistic and p-value are two essential tools for hypothesis testing that determine if observed differences in data are statistically significant [39]. By comparing the observed difference to the predicted variation under the null hypothesis, the t-statistic determines if the means of the two groups are statistically different. If the absolute t-statistic is large, the observed difference is unlikely to have occurred by chance. This is complemented by the p-value, which measures the likelihood of receiving an effect that is at least as extreme as the one actually observed, assuming that the null hypothesis is correct. Strong evidence against the null hypothesis is shown by a lower p-value, which is usually less than 0.05 and denotes the statistical significance of the observed differences. These statistical metrics are essential for verifying that the reported gains are statistically significant and not the result of chance when comparing the performance of SGD, ASGD, and CSGD. The t-statistics and p-values presented in Table 14 demonstrate that the improvement in ASGD over conventional SGD is statistically significant. Along with the corresponding t-statistics and p-values, the table presents important performance metrics like accuracy, precision, recall, F1-score, and execution time. The accuracy of ASGD indicates a significant improvement with a t-statistic of -15.0570 and a p-value of 0.0000. With a t-statistic of -29.0000, precision shows the largest change, confirming ASGD's capacity to improve the identification of fraudulent cases. With a t-statistic of -17.6756 for the recall metric, the increased sensitivity in identifying fraudulent cases is evident. Furthermore, the F1 score, which strikes a balance between recall and precision, displays a t-statistic of -15.9217, indicating a strong improvement in overall performance. The t-statistic of -25.2983 indicates a significant difference in computational efficiency and execution time is also significantly impacted. The efficacy of ASGD in

Table 14
Paired t-test results for ASGD and CSGD

Metric	ASGD t-Statistic	ASGD p-Value	CSGD t-Statistic	CSGD p-Value
Accuracy	-15.0570	0.0000	-9.4868	0.0000
Precision	-29.0000	0.0000	-9.7838	0.0000
Recall	-17.6756	0.0000	-3.3017	0.0092
F1-score	-15.9217	0.0000	-7.1804	0.0001
Execution Time	-25.2983	0.0000	-8.8345	0.0000

maximizing classification performance is validated by the consistently low p-values (all below 0.05), which verify that these improvements are statistically significant. The fundamental formula that drives ASGD's weight update is:

$$\theta \leftarrow \theta - \eta \cdot \nabla_\theta \mathcal{L}, \tag{22}$$

where θ represents the model parameters being updated, η is the learning rate that controls the step size of the update, $\nabla_\theta \mathcal{L}$ denotes the gradient of the loss function $\mathcal{L}$ with respect to θ, and $\theta \leftarrow \theta - \eta \cdot \nabla_\theta \mathcal{L}$ describes the parameter update rule in gradient-based optimization.
The statistical analysis in Table 14 shows that CSGD significantly outperforms traditional SGD. Through the integration of CS optimization with SGD, CSGD gains access to both local gradient-based refinement and global search capabilities. With a t-statistic of -9.4868 and a p-value of 0.0000, the accuracy metric shows a significant improvement. With a t-statistic of -9.7838, precision increases noticeably, demonstrating the model's capacity to lower misclassification errors. The recall metric shows an enhanced capacity to identify fraudulent cases with a t-statistic of -3.3017 and a p-value of 0.0092. The robustness of the CSGD model is validated by the F1-score, which strikes a balance between precision and recall with a t-statistic of -7.1804. Moreover, a t-statistic of -8.8345 for execution time indicates a significant influence on computational efficiency. The efficacy of CSGD in maximizing classification performance is demonstrated by the consistently low p-values (all below 0.05), which confirm that these performance gains are statistically significant. The CSGD optimization procedure is based on the following equation:

$$x_i^{\text{new}} = x_i + \beta \cdot L(\lambda), \tag{23}$$

where $L(\lambda)$ denotes the Lévy flight distribution, x_i is the current solution, and β is the step size scaling factor. CSGD can better explore the solution space and break out of local minima due to the global search method.

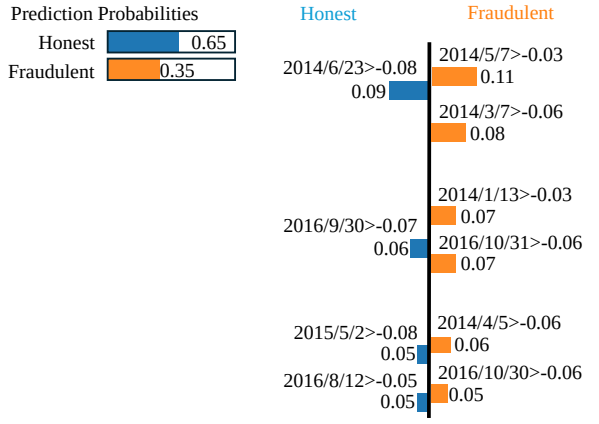
4.9. LIME: Enhancing Interpretability of ASGD and CSGD Proposed Models

LIME is developed to help in understanding the predictions made by complex ML and DL models. By producing locally interpretable explanations, it aims to explain individual predictions for particular data points rather than the model predictions as a whole. By building a more straightforward interpretable model around the neighborhood of each prediction, LIME enables us to determine which features have the highest influence [40]. This method makes it particularly helpful when dealing with black-box ASGD and CSGD models, which are extremely accurate but challenging to understand. For the models to generate predictions on the modified versions, it first slightly alters the instance that we want to explain and produce numerous similar instances. LIME employs these predictions to learn a straightforward, interpretable model in the vicinity of the initial data point. By locally simulating the behavior of the complex model, the simpler model demonstrates which features had the highest influence on a particular prediction. When we concentrate on the feature that LIME selected, it becomes clear why the proposed models generated particular predictions [41].

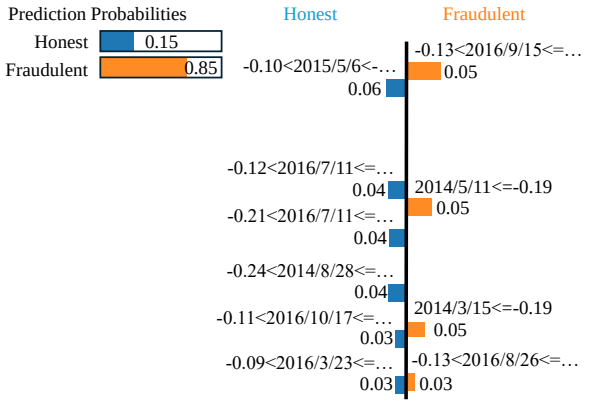
In this section, we present a case study based on two specific instances selected from the ASGD and CSGD models to demonstrate the use of XAI techniques, specifically LIME. The goal is to show how LIME decodes the decision-making process of these models in ETD. For a better understanding, we analyze two instances with different influential features. The first instance is primarily influenced by a normal user, indicating unusual energy consumption at specific times. The second instance is influenced by irregular load patterns, highlighting different aspects of ET behavior. By applying LIME, we are able to identify the features that had the greatest impact on the model's prediction, thus providing insights into how the ASGD and CSGD models make their decisions. The probability prediction and summary plot explanations are shown in Figures 15 and 16, respectively.

The Figure 15a presented here illustrates the LIME explanation for the ASGD model's prediction for 200^{th} instance. This instance was predicted as honest with a probability of 0.65 for the *Honest* class and 0.35 for the *Fraudulent* class. This prediction suggests that the ASGD model has moderate confidence in classifying the instance as honest.

Figure 15a shows the contributions of various features, represented along the y-axis. The x-axis indicates the magnitude of each feature's contribution to the final prediction. Negative values (blue bars) contribute towards the *Honest* prediction, while positive values (orange bars) contribute towards the *Fraudulent* prediction. The longer the bar, the



(a) ASGD prediction for 200^{th} instance for electricity theft detection



(b) CSGD prediction with 400^{th} instance for electricity theft detection with LIME

Figure 15: Comparison of ASGD and CSGD predictions for electricity theft detection on different instances.

more influential that feature is in determining the final classification.

From the plot, the following key observations can be made: the feature $2014/6/23 > -0.08$ with value of +0.09 contributes significantly to the *Honest* prediction. The negative value indicates that the consumption pattern on 2014/6/23 was consistent with regular, non-fraudulent behavior. The absence of any sudden drops or abnormal spikes in consumption suggests legitimate electricity usage. Another feature $2016/9/30 > -0.07$ with value of +0.06: the feature also contributes negatively to the *Honest* classification. The consumption pattern during this period is stable and regular, reinforcing the model's decision that the instance is honest. Similarly, features 2015/5/2 > -0.08 and 2016/8/12 > -0.05 with values of +0.05; these feature also reflects a stable consumption period. The ASGD model associates this with an honest classification, as there are no irregularities in usage.

On the other hand, feature $2014/5/7 > -0.03$ with the value of -0.11 contributes significantly to the *Fraudulent* classification. The positive contribution indicates that consumption during this period deviated from the norm, possibly indicating suspicious or fraudulent behavior. Similarly, features $2014/1/13 > -0.03$ and $2016/10/31 > -0.06$ with values of -0.07 contribute positively, suggesting that consumption during this period was anomalous, further supporting the fraudulent classification. Moreover, feature $2014/3/7 > -0.06$ and 2014/4/5 > -0.06 with values of -0.08 and -0.06 contribute to the *Fraudulent* prediction. The model notes irregular consumption during this time, contributing to a higher likelihood of fraud.

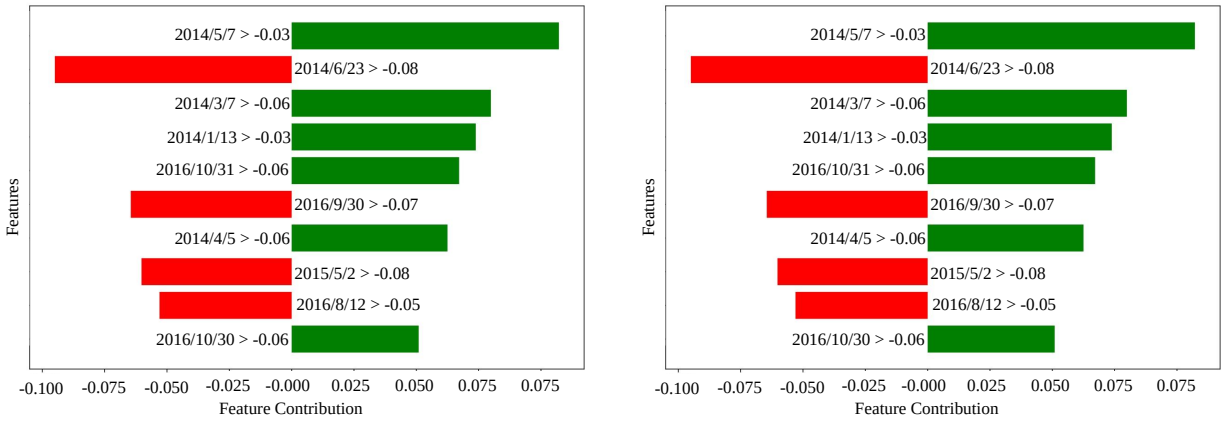
In conclusion, the LIME explanation plot for this instance demonstrates that the *Honest* prediction is driven by several features that show stable, consistent consumption, such as $2014/6/23 > -0.08$ and $2016/9/30 > -0.07$, which contribute negatively to the final classification. However, there are also features, such as $2014/5/7 > -0.03$ and $2014/3/7 > -0.06$, that contribute positively, suggesting periods of irregular consumption that lean towards the *Fraudulent* class.

The model's prediction of *Honest* usage with a probability of 0.65 indicates that while there are some signs of irregular consumption, the majority of features indicate normal usage. The LIME explanation helps us understand how individual features contribute to the prediction, providing transparency into the model's decision-making process and enabling stakeholders to interpret the ASGD model's predictions more effectively. This transparency is crucial for validating the ASGD model's predictions and making informed decisions in real-world applications such as ETD, where understanding the influence of each feature is key to improving the model's reliability.

The Figure 15b presented here illustrates the LIME explanation for the CSGD model's prediction for the 400^{th} instance in the dataset. In this case, the instance was predicted to be fraudulent with a high probability of 0.85 for the *Fraudulent* class and 0.15 for the *Honest* class, indicating that the CSGD model was confident in its classification.

Figure 15b shows the contributions of various features to the CSGD model's decision, with the $x - axis$ representing the magnitude of the feature contributions. Positive values (blue bars) push the prediction towards *Honest*, while negative values (orange bars) push it towards *Fraudulent*. The $y - axis$ displays the features, which are time-based consumption intervals. The magnitude of the bars shows the level of influence each feature had on the final prediction.

From the LIME explanation, we can observe that several features contributed positively to the *Honest* classification, where he feature $-0.10 < 2015/5/6 <= ...$ contributed a positive value of +0.06, indicating that this consumption pattern was typical and not indicative of fraudulent activity. Similarly, the feature $-0.12 < 2016/7/11 <= ...$ had a positive contribution of +0.04, suggesting that stable usage during this period reinforced the CSGD model's prediction



(a) Summary plot of ASGD prediction for 200^{th} instance in electricity theft detection

(b) Summary plot for CSGD prediction for 400^{th} instance in electricity theft detection

Figure 16: Comparison of summary plots for ASGD and CSGD predictions in electricity theft detection.

of honest behavior, supporting the conclusion that the consumption pattern during this period was normal and legitimate. Moreover, features such as $-0.24 < 2014/8/28 <= ...$ and $-0.11 < 2016/10/17 <= ...$ had smaller but still positive contributions of +0.04 and +0.03, respectively, indicating that regular consumption over these periods helped the model classify the instance as honest. Finally, the feature $-0.09 < 2016/3/23 <= ...$ contributed a small positive value of +0.03, further supporting the prediction of honest consumption.

On the other hand, there were a few features that contributed negatively to the prediction, the feature $-0.13 < 2016/9/15 <= ...$ contributed a small negative value of -0.05, slightly favoring the *Fraudulent* prediction. However, this was not strong enough to overturn the model's overall classification of *Fraudulent*. Similarly, the feature $2014/5/11 <= -0.19$ also had a negative contribution of -0.05, but its impact was increased. Another feature, $-0.13 < 2016/8/26 <= ...$, contributed a negative value of -0.03, which, although negative, was still significant enough to affect the final prediction.

In conclusion, the LIME plot for the 400th instance reveals that the majority of features contributed negatively to the *Fraudulent* classification, with fluctuating and unpredictable consumption patterns being the main factors. Features like $-0.13 < 2016/9/15 <= ...$ and $-0.19 < 2014/5/11 <= ...$ were key contributors, with the model assigning a high confidence score of 0.85 to the *Fraudulent* prediction. The positive contributions from features like $-0.10 < 2015/5/6 <= ...$ and $2016/7/11 <= -0.12$ had only a minimal impact, not enough to change the CSGD model's decision.

This explanation provides transparency into the model's reasoning, helping stakeholders understand which features influence the model's predictions and how they contribute to the classification of legitimate electricity usage.

The Figure 16a presented here illustrates the LIME explanation for the ASGD model's prediction for the 200th instance in the dataset. The LIME plot shows the contributions of various features, where positive values (indicated by green bars) contribute towards the *Honest* class, and negative values (indicated by red bars) contribute towards the *Honest* class.

The $x - axis$ in the Figure 16a represents the magnitude of each feature's contribution, and the $y - axis$ lists the time-based features corresponding to different consumption intervals. By examining this plot, we can understand the individual contributions of various consumption features towards the ASGD model's decision.

The feature $2014/5/7 > -0.03$ with value if 0.09 contributes positively to the prediction, suggesting that the consumption pattern during this time period is inconsistent with irregular and illegitimate usage. The ASGD model interprets this feature as indicative of *Fraudulent* behavior, and its positive contribution reinforces the fraudulent classification. In contrast, feature $2014/6/23 > -0.08$ with value of 0.09 corresponding to consumption during 2014/6/23, contributes negatively to the *Honest* prediction. The ASGD model associates stable, non-anomalous usage with legitimate consumption. Moreover, features $2014/1/13 > -0.03$, $2016/10/31 > -0.06$, and $2014/3/7 > -0.06$ with values of 0.07, 0.07, and 0.08 reflect unstable consumption periods that positively support the *Fraudulent* classification. Consumption data from these features also contributes positively, suggesting irregular usage, which the ASGD model associates with fraudulent behavior. Finally, features $2016/9/30 > -0.07$, $2015/5/2 > -0.08$, and $2016/8/12 > -0.05$ with values of 0.06, 0.05, and 0.05 contributes negatively to the ASGD model's decision, suggesting another regularity in the consumption pattern that the model associates with honest behavior.

The LIME explanation for the 200^{th} instance in the ASGD model shows that the *Honest* classification is primarily driven by features such as $2014/6/23 > -0.030$, $2016/9/30 > -0.06$, and $2015/5/2 > -0.05$, which all contribute negatively to the final prediction. These features

reflect stable and consistent electricity consumption, which the model interprets as legitimate usage.

Although there are positive contributions from features like $2014/5/7 > -0.11$ and $2014/3/7 > -0.08$, these irregularities do not overpower the negative contributions from stable usage features. The overall prediction remains *Honest*, with the majority of the contributing features indicating regular consumption patterns. The LIME explanation helps to clarify the ASGD model's decision-making process by providing insight into how specific features, such as stable consumption periods, influence the classification of honest electricity usage. This transparency is crucial for understanding and interpreting the ASGD model's predictions, especially in the context of detecting ET.

The Figure 16b presented here shows the LIME explanation for the CSGD model's prediction for the 400th instance in the dataset. The plot represents the feature contributions to the CSGD model's prediction, where positive values (indicated by green bars) push the prediction towards the *Fraudulent* class, and negative values (indicated by red bars) push it towards the *Honest* class.

From the plot, we observe the following features contributing to the honest prediction, the feature $-0.10 < 2015/5/6 <= ...$ shows a negative contribution of +0.08 towards the honest prediction. This suggests that the consumption pattern during this period is consistent with normal, non-fraudulent usage. Similarly, the feature $2016/7/11 <= -0.19$ contributed a negative value of +0.05, indicating normal usage during this time period, reinforcing the prediction of honest behavior. Another feature, $-0.11 < 2016/10/17 <= ...$ also had a positive contribution of +0.07. This feature reflects consistent usage without deviations, further supporting the honest classification. Additionally, the features $-0.09 < 2016/3/23 <= ...$ and $-0.24 < 2014/8/28 <= ...$ also had small positive contributions, with values of +0.04 and +0.10, respectively. These features show regular consumption during the specified periods, which the model associates with legitimate electricity usage.

On the other hand, several features contributed positively to the fraudulent classification, the feature $-0.13 < 2016/9/15 <= ...$ contributed negatively with a value of -0.03. This suggests a slight deviation from the expected consumption pattern during this period, though its impact on the final classification was higher. The feature $-0.19 < 2016/5/11 <= ...$ had a larger negative contribution of -0.06, indicating some irregularities in the consumption behavior during this period. However, this feature's negative contribution was strong enough to change the overall prediction to the fraudulent class. Several other features, such as $-0.19 < 2014/3/15 <= ...$ and $-0.13 < 2016/8/26 <= ...$, also contributed negatively, with values of -0.06 and -0.04, respectively. Despite these negative impacts, the overall contribution of the positive features was stronger, leading to the final prediction of fraudulent usage.

In conclusion, the LIME plot for the 400th instance demonstrates that the CSGD model's classification of fraudulent electricity usage is primarily driven by unstable and unpredictable consumption patterns, such as those observed in $-0.13 < 2016/9/15 <= ...$ and $2014/5/11 <= -0.19$. These features contributed positively to the fraudulent classification. While a few features, like $-0.10 < 2015/5/6 <= ...$ and $-0.12 < 2016/7/11 <= ...$, contributed negatively, their effects were minimal compared to the overall positive contributions. This explains why the CSGD model classified the instance as fraudulent with a high confidence of 0.85. The LIME explanation helps to understand how individual features influenced the CSGD model's decision and provides transparency into the decision-making process.

4.10. SHAP: Enhancing Explainability of ASGD and CSGD Proposed Models

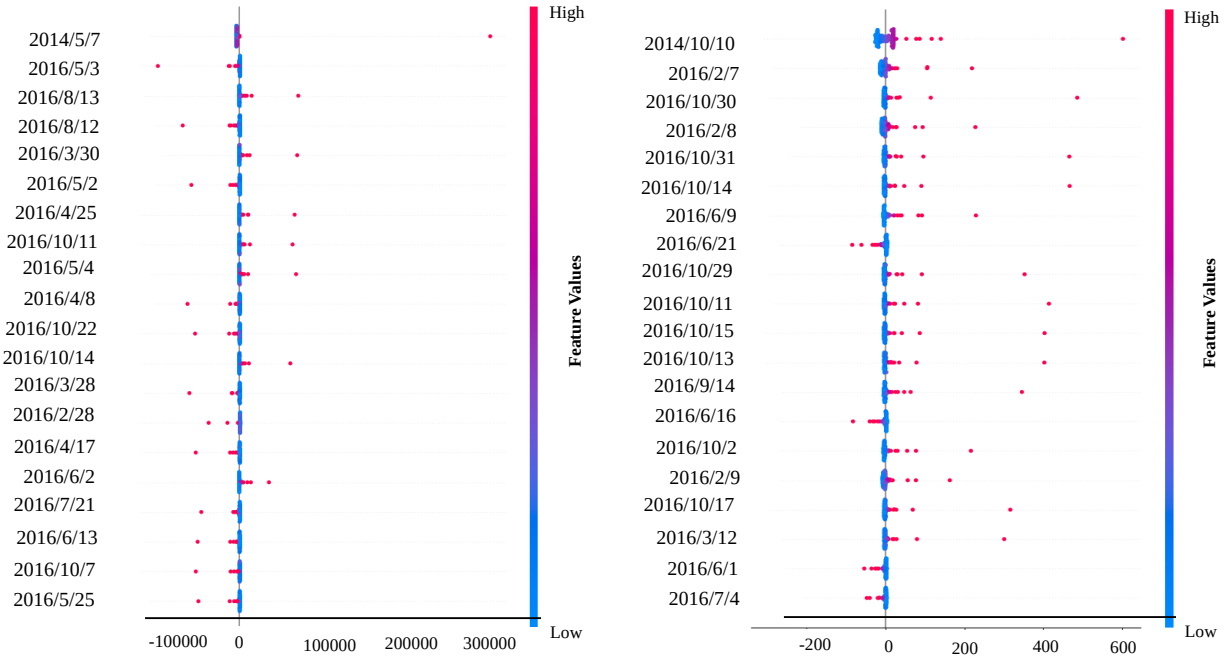
The objective of the cooperative game theory of SHAP is to fairly allocate the prediction among the features. SHAP uses Shapley values, which come from the idea of equitable distribution in cooperative games. For a feature i, the Shapley value is calculated by taking into account every possible feature combination and assessing the feature contributions to the output [42]. Mathematically, Shapley value for the feature i is calculated using:

$$\phi_i(f) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} [f(S \cup \{i\}) - f(S)], \quad (24)$$

where $\phi_i(f)$ denotes the Shapley value of feature i for the model f, N is the set of all features, S represents a subset of features excluding feature i, $f(S)$ represents the model's prediction when just the features from subset S are present, $f(S \cup \{i\})$ is the ASGD and CSGD model's prediction when the feature i is added to the subset S, $|S|$ is the number of features in subset S, and $|N|$ denotes the total number of features. Each feature's contribution is assessed across all potential feature combinations, providing a fair and consistent explanation of how each feature affects the model's output. SHAP's additive nature ensures that the difference between both models' predicted and expected value of the output (often referred to as the base value) is equal to the sum of the Shapley values of all features using:

$$f(x) = \phi_0 + \sum_{i=1}^{|N|} \phi_i(f) \quad (25)$$

where ϕ_0 is the base value, and the difference between the prediction and the base value is equal to the sum of the Shapley values for all features. By breaking down the prediction into contributions from each feature, SHAP allows not only to comprehend the relative importance of each feature but also provides a clear and comprehensive picture of the ASGD and CSGD model's behavior. Figures 17, 18, and 19, present the beeswarm plot, waterfall plot, and force plot for ASGD and CSGD, respectively, which are used to analyze the feature importance, model prediction behavior, and decision-making process both locally and globally in ETD.



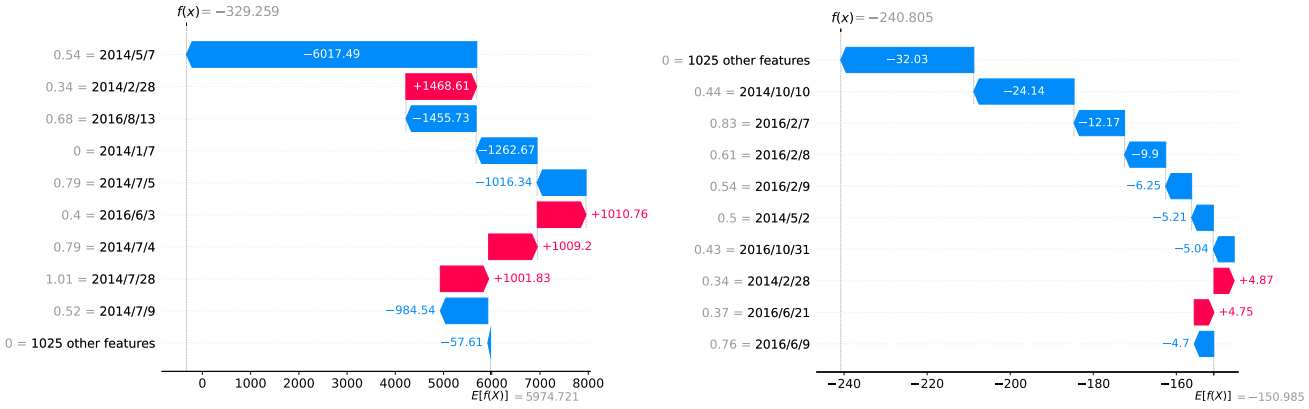
(a) Beeswarm plot for ASGD in electricity theft detection

(b) Beeswarm plot for CSGD in electricity theft detection

Figure 17: Explanations of beeswarm plots for ASGD and CSGD predictions in electricity theft detection.

The beeswarm plot shown in Figure 17a is used to display the ASGD model SHAP values. The $x-axis$ represents the different features, such as 2014/5/7, 2016/5/3, and 2016/8/13, etc., while the $y-axis$ shows the range of feature values, with the plot providing insights into the spread and density of feature values across these features. The plot covers a range of values for each time interval, with some features showing relatively high concentrations (e.g., 2014/5/7), while others have a more uniform or less concentrated distribution (e.g., 2016/5/25). The distribution appears to vary across features, which may reflect fluctuations in electricity consumption patterns over time, with certain periods showing more regular or predictable consumption, while others display more variability. The 2016/5/3 feature values for this period are concentrated in the middle range of the y-axis, suggesting that consumption during this time was relatively stable. The model might consider such stable consumption periods as indicators of legitimate (honest) usage. Similarly, feature 2016/8/13 shows a broader distribution of values, with some values extending toward both high and low ends of the scale. This might indicate periods of irregular consumption that could be indicative of fraudulent activity, as a broader range of consumption values often suggests irregularities in usage patterns. In contrast, 2016/10/11 feature values here appear more concentrated in the lower range, suggesting lower-than-usual consumption during this period. This could indicate reduced consumption, possibly due to external factors like seasonal variations, which the ASGD model may interpret as either normal or indicative of honest usage. By examining the distribution of feature values across time intervals, the ASGD model gains a better understanding of typical usage patterns and can more accurately classify electricity consumption as honest or fraudulent. The beeswarm plot serves as a valuable tool for interpreting the spread of feature values and their relationship with the final predictions made by the ASGD model.

The Figure 17b presented here illustrates the SHAP beeswarm plot for the CSGD model's predictions. This plot shows the distribution of SHAP values for each feature over all instances, which helps in understanding how each feature impacts the model's predictions across the entire dataset. The y-axis lists the features corresponding to different time-based intervals, such as 2014/10/10, 2016/2/7, and 2016/10/30. These represent various time periods of electricity consumption. The $x-axis$ represents the SHAP values, which quantify the impact of each feature on the CSGD model's output. Negative SHAP values (indicated by the red part of the plot) push the prediction towards higher values (likely indicating honest behavior), while positive SHAP values (shown by the blue part) push it towards lower values (likely indicating fraudulent behavior). Each point along the $x-axis$ represents the SHAP value of a specific feature for a given instance. The color gradient from blue (low feature values) to red (high feature values) indicates the range of feature values for each feature, providing insight into how different levels of a feature influence the CSGD model's predictions. Features including 2014/10/10, 2016/10/14, 2016/6/9, etc., are located towards the lower range of SHAP values, indicating that it predominantly influences the prediction towards $Honest$ behavior. The distribution of points for this feature



(a) Waterfall plot for ASGD in electricity theft detection

(b) Waterfall plot for CSGD in electricity theft detection

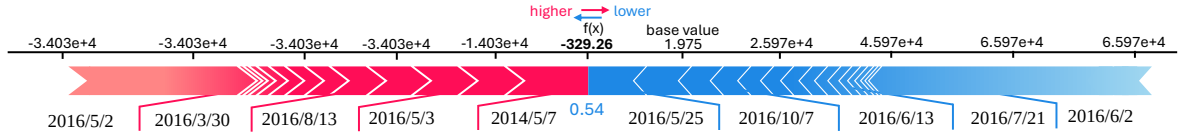
Figure 18: Comparison of waterfall plots for ASGD and CSGD predictions in electricity theft detection.

is centered around negative SHAP values, reflecting that the CSGD model considers the consumption pattern during this period. These features also show distributions of SHAP values that lean more towards negative values, suggesting that the CSGD model associates these periods with more typical, honest behavior. In contrast, features including 2016/2/7, 2016/10/30, 2016/10/31, etc., have a relatively balanced distribution, with both positive and negative SHAP values. This indicates that the CSGD model associates this feature with both *Honest* and *Fraudulent* behavior, depending on the consumption context. These features show a wide spread of SHAP values, with points scattered across both negative and positive values. This suggests that the model sees these time periods as more ambiguous or possibly irregular, indicating a mix of honest and fraudulent patterns depending on the specific instances of consumption.

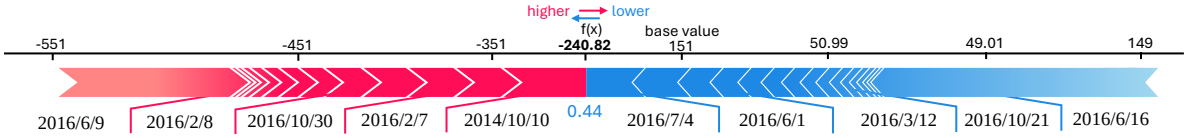
The waterfall plot in Figure 18a shows the way various features affect the ASGD model's predictions. A feature's importance to the model's decision-making process is indicated by the intensity of each bar that reflects that feature. Increased bar features have a major impact on whether an instance is categorized as fraudulent or honest. The ASGD model's predictions are easier to understand due to the waterfall plot; it also demonstrates the most crucial features for classification, and the ASGD AL strategy gives priority to difficult and uncertain instances. Features like 2014/2/28, 2016/6/3, 2014/7/4, and 2014/7/28 are the most contributing features from the negative (honest) class, have the highest SHAP values, and favorably impact the ASGD model's classification. Similarly, features like 2014/5/14, 2016/8/13, 2014/1/7, and 2014/7/5 are the most contributing features from the positive fraudulent class. Furthermore, the prediction is reinforced by the significant contributions made by 2014/5/7 and 2016/8/13. In contrast, the features that contribute the least, like 2014/7/5 and 2014/7/9, have little bearing on the choice made in the end and barely affect the ASGD model's prediction. The cumulative sum of these contributions results in a final prediction value of 5974.721, with the model's output function $f(x) = 329.259$. This final prediction value suggests that the instance is most likely classified as *Honest*.

A waterfall plot for the CSGD model is presented in Figure 18b. The $x-axis$ displays different dataset features, while the $y-axis$ shows the mean absolute SHAP values. To influence the CSGD model predictions, which are essential for identifying ET, highlights the features according to their significance. The degree to which a feature influences the model's output is indicated by each bar's length and its values. The plot shows that the most important features, those with higher SHAP values, have a greater impact on identifying whether an incident is considered fraudulent or honest. 2016/2/7, 2014/10/10, and 2016/10/31 are the most contributing features with the highest SHAP values that have a major effect on the classification. On the other hand, features with lower SHAP values, like 2016/6/21 and 2014/2/28, have little bearing on the choice. The CSGD model makes use of these high-impact features to produce predictions that are more accurate due to its effective data handling and strong learning methodology. CSGD enhances decision boundaries, particularly in complex and imbalanced datasets like those used for ETD, by concentrating on the most illuminating instances. The positive contributions from features such as 2014/2/28 with (+32.03) and 2016/10/21 with (+24.14) push the prediction toward the honest classification, reinforcing the CSGD model's decision. The final prediction value is $E[f(X)] = 150.985$, which is derived from these contributions. The waterfall plot provides a clear visualization of how the ASGD model arrives at its final prediction by showing the impact of each feature.

Figure 19a displays a force plot for utilizing ASGD for ETD. The plot visually explains the ASGD model's classification of the specific 1^{st} instance. The SHAP values assigned to each feature demonstrate the individual feature's effect on the final prediction. Features are ranked based on the feature contribution to the ASGD model's output. The base value on the plot is -329.26, representing the initial or default prediction before considering any features.



(a) Force plot of class 0 for ASGD in electricity theft detection



(b) Force plot of class 0 for CSGD in electricity theft detection

Figure 19: Comparison of force plots for class 0 of ASGD and CSGD in electricity theft detection.

Each feature then pushes this base value either higher or lower, with positive values contributing to a higher predicted output, and negative values pushing the prediction lower. The final predicted value of the instance is shown to be around 6.597e+4. This final prediction is reached by summing the contributions of various features over time. Features including 2014/5/7, 2016/5/3, 2016/8/13, 2016/3/30, and 2016/5/2 have negative contributions towards the ASGD model's final prediction, pushing the value slightly higher. This indicates that the model saw this period as having legitimate usage, contributing positively to the classification. In contrast, features including 2016/5/25, 2016/10/16, 2016/6/13, 2016/7/21, and 2016/6/2 have significant positive contributions, further increasing the prediction. This reinforces the decision towards fraudulent usage during these periods. The final prediction is shown to be approximately 6.597e+4, which reflects the cumulative effect of all the features. The force plot helps to visualize how each feature impacts the final prediction for the given 1^{st} instance. This transparency into the model's decision-making process allows for a better understanding and interpretation of how individual consumption features influence the overall prediction of electricity usage.

The Figure 19b presented here illustrates the force plot for the CSGD model's prediction for a 0^{th} instance. Force plots visualize how individual features push the model's prediction higher or lower relative to a base value. In this case, the base value is -240.82, representing the starting point before any feature contributions are considered. The force plot shows how each feature influences the final prediction. Positive contributions (shown in blue) push the prediction higher, while negative contributions (shown in red) pull the prediction lower. Features including 2014/10/10, 2016/2/7, 2016/10/30, 2016/2/8, and 2016/6/9 have negative contributions towards the CSGD model's final prediction, pushing the value slightly higher. This indicates that the model saw this period as having legitimate usage, contributing positively to the classification. In contrast, features including 2014/7/4, 2016/6/1, 2016/3/12, 2016/10/21, and 2016/6/16 have significant positive contributions, further increasing the prediction. The force plot provides valuable insights into how individual features influence the model's predictions. Stable, predictable consumption patterns, such as those observed on 2014/10/10 and 2016/2/7, contribute positively to the prediction, while features like 2016/2/8 and 2016/6/9 with irregular consumption patterns pull the prediction downward, indicating potential fraud. The force plot effectively visualizes the model's reasoning by illustrating how the individual time-based consumption features push the final decision towards either honest or fraudulent classifications.

In cases where SHAP and LIME provide conflicting feature importance rankings, stakeholders should understand the difference between the two techniques: SHAP offers a global explanation based on the overall behavior of the model, while LIME provides a local explanation, focusing on specific predictions. If SHAP ranks a feature as highly important while LIME does not, this can indicate that the feature plays an important role across the dataset but may not be influential in specific, local predictions. Stakeholders should consider the context when interpreting these discrepancies, taking into account whether a global or local understanding of feature importance is more critical for their needs.

4.11. Critical Analysis of LIME and SHAP

Two popular explainable AI techniques are LIME and SHAP. SHAP seeks to increase the transparency of ML models by offering explanations for their predictions. However, there are significant differences between their underlying methodologies, scope, and reliability. SHAP addresses some of LIME's inherent limitations. LIME is an interpretable model that usually uses LR to locally approximate the decision boundary of a complex model to explain individual predictions. By generating slightly altered versions of the original instance, it modifies the input data and tracks the resulting changes in the model's output. By giving features importance weights based on these perturbations, LIME effectively identifies the features that had the highest impact on the prediction. LIME has several drawbacks. If the method is repeated on the same data point, its dependence on random alterations to the

provided data yields inconsistent outcomes. The local area surrounding the instance is not always linear, particularly if the decision surface of the model is extremely non-linear. Furthermore, user-defined parameters, like the kernel function and distance metrics, significantly affect the way in which LIME explains things, and choosing them isn't always obvious. Due to the need to create and assess several perturbations for every explanation, the method is also computationally costly.

However, SHAP offers a theoretically supported framework for determining each feature's contributions to a prediction by expanding on the idea of Shapley values from cooperative game theory. It ensures that feature attributions are fair, consistent, and interpretable globally by taking into account all potential feature subsets when calculating their marginal contribution. In contrast to LIME, SHAP gives a global interpretability of feature importance for the complete dataset in addition to local explanations for particular instances. Several LIME drawbacks are addressed by SHAP. Because it depends on Shapley values, feature attributions are reliable and consistent. For example, if a model is modified in a way that increases the impact of a feature, SHAP will appropriately reflect this change. It can also handle a variety of model types, such as ensemble methods, neural networks, and tree-based models, by combining multiple interpretability techniques into a single framework.

Both techniques improve the black-box models' interpretability, making predictions that stakeholders, like utility companies, can comprehend. SHAP provides insight into specific features, like daily patterns of electricity consumption, affecting the predictions feature, and LIME assists in validating specific theft cases. SHAP determines which features are most important in differentiating between theft and honest cases. This aids in improving feature engineering procedures, which improves the proposed models' performance.

4.12. Relationship between LIME, SHAP, and Influential Components of the Proposed Framework

In response to Reviewer 5's concern regarding the relationship between the interpretable AI methods LIME and SHAP and the other components of the proposed framework, we further elaborate on their roles. LIME and SHAP are both integral to the interpretability and transparency of the ASGD and CSGD models in ETD.

LIME is used to provide local interpretability for individual predictions made by the ASGD and CSGD models. By approximating the decision boundary around specific data points, LIME enables stakeholders to understand why the model classifies certain instances as fraudulent or honest, making the model's decisions more transparent to stakeholders. This localized explanation is particularly beneficial in ETD, where identifying the reasoning behind individual predictions is crucial for validation and trust.

SHAP, on the other hand, offers both local and global interpretability. In our framework, SHAP is used to analyze the global feature importance across the entire dataset, helping to understand which features contribute most to detecting ET. Additionally, SHAP values are used to track the influence of each feature in the decision-making process of ASGD and CSGD, providing a comprehensive overview of the model's behavior and ensuring that the results are interpretable across all instances.

These interpretability methods are aligned with the framework's goal of ensuring fairness, transparency, and trust in the model's decisions. The integration of LIME and SHAP with the ASGD and CSGD models enhances their usability, making them more accessible to utility companies, regulators, and other stakeholders who require an understanding of the models' inner workings to act on fraud detection outcomes confidently. Moreover, these methods complement the optimization techniques used in the models, such as AL and optimization algorithms, by providing transparency on how these techniques contribute to the final predictions.

Thus, the interpretability methods LIME and SHAP are not only crucial for enhancing the trustworthiness of the proposed ETD models but also closely relate to the overall optimization framework that includes the use of AL and optimization techniques in improving the model's accuracy, efficiency, and performance. This dual approach of providing both local and global interpretability helps mitigate the challenges of data imbalance and computational efficiency, which are pivotal in real-world ETD applications.

5. Conclusion and Future Work

In this study, novel methods for ETD were proposed using the highly imbalanced SGCC dataset. The LoRAS oversampling technique is utilized to balance the dataset, ensuring equitable representation of both majority and minority classes. To enhance classification performance, we proposed two novel models, ASGD and CSGD. ASGD integrates SGD and entropy-based AL, which improves classification boundaries and produces better results by giving priority to uncertain instances. ASGD outperformed the conventional model by achieving an accuracy of 0.82, an F1-score of 0.83, and PR-AUC of 0.91. Similarly, CSGD combines CS with SGD to efficiently explore parameter spaces and adjust to non-linear relationships, resulting in high recall and precision. CSGD's strong classification ability is demonstrated by an accuracy of 0.81, an F1-score of 0.82, and PR-AUC of 0.90. The performance of the ASGD and CSGD models was validated using 10-FCV, ensuring a robust evaluation. The statistical significance of ASGD and CSGD was confirmed using a t-test. Furthermore, LIME and SHAP, two XAI techniques, were used to evaluate the interpretability and explainability of the proposed models. These methods validate model decisions, improve transparency, and shed light on the significance of features.

In the future, we can enhance the performance of the proposed models by utilizing advanced data augmentation, feature engineering, and transfer learning techniques. The model's robustness and generalizability will also be assessed by applying them to a variety of datasets.

References

[1] Ballal, M.S., Suryawanshi, H., Mishra, M.K. and Jaiswal, G., 2020. Online electricity theft detection and prevention scheme for smart cities. IET Smart Cities, 2(3), pp.155-164.

[2] Shahzadi, N., Javaid, N., Akbar, M., Aldegheishem, A., Alrajeh, N. and Bouk, S.H., 2024. A novel data driven approach for combating energy theft in urbanized smart grids using artificial intelligence. Expert Systems with Applications, 253, p.124182.

[3] Wen, H., Liu, X., Lei, B., Yang, M., Cheng, X. and Chen, Z., 2025. A privacy-preserving heterogeneous federated learning framework with class imbalance learning for electricity theft detection. Applied Energy, 378, p.124789.

[4] Ullah, K., Hafeez, G., Khan, I., Jan, S. and Javaid, N., 2021. A multi-objective energy optimization in smart grid with high penetration of renewable energy sources. Applied Energy, 299, p.117104.

[5] Khan, I.U., Javeid, N., Taylor, C.J., Gamage, K.A. and Ma, X., 2021. A stacked machine and deep learning-based approach for analysing electricity theft in smart grids. IEEE Transactions on Smart Grid, 13(2), pp.1633-1644.

[6] Wang, Y., Yu, T., Luo, Q., Liu, X., Wang, Z., Wu, Y. and Pan, Z., 2024. Two-stage generalizable approach for electricity theft detection in new regions. Applied Energy, 365, p.123228.

[7] Ammar, M., Javaid, N. & Arishi, A., 2025. An AI Explained Systematic Modular Approach for Enhanced Electricity Theft Detection for Urbanized Smart Grid Environment. Alexandria Engineering Journal.

[8] Razavi, R., Gharipour, A., Fleury, M. and Akpan, I.J., 2019. A practical feature-engineering framework for electricity theft detection in smart grids. Applied energy, 238, pp.481-494.

[9] Hussain, S., Mustafa, M.W., Jumani, T.A., Baloch, S.K., Alotaibi, H., Khan, I. and Khan, A., 2021. A novel feature engineered-CatBoost-based supervised machine learning framework for electricity theft detection. Energy Reports, 7, pp.4425-4436.

[10] Gunturi, S.K. and Sarkar, D., 2021. Ensemble machine learning models for the detection of energy theft. Electric Power Systems Research, 192, p.106904.

[11] Tsao, Y.C., Rahmalia, D. and Lu, J.C., 2024. Machine-learning techniques for enhancing electricity theft detection considering transformer reliability and supply interruptions. Energy Reports, 12, pp.3048-3064.

[12] Abbas, S., Bouazzi, I., Ojo, S., Sampedro, G.A., Almadhor, A.S., Al Hejaili, A. and Stolicna, Z., 2023. Improving smart grids security: An active learning approach for smart grid-based energy theft detection. IEEE Access, 12, pp.1706-1717.

[13] Kong, X., Zhao, X., Liu, C., Li, Q., Dong, D. and Li, Y., 2021. Electricity theft detection in low-voltage stations based on similarity measure and DT-KSVM. International Journal of Electrical Power & Energy Systems, 125, p.106544.

[14] Echard, B., Gayton, N. and Lemaire, M., 2011. AK-MCS: an active learning reliability method combining Kriging and Monte Carlo simulation. Structural safety, 33(2), pp.145-154.

[15] Lepolesa, L.J., Achari, S. and Cheng, L., 2022. Electricity theft detection in smart grids based on deep neural network. Ieee Access, 10, pp.39638-39655.

[16] Sarker, M.A.A., Shanmugam, B., Azam, S. and Thennadil, S., 2024. Enhancing smart grid load forecasting: An attention-based deep learning model integrated with federated learning and XAI for security and interpretability. Intelligent Systems with Applications, 23, p.200422.

[17] Liao, W., Yang, Z., Liu, K., Zhang, B., Chen, X. and Song, R., 2022. Electricity theft detection using Euclidean and graph convolutional neural networks. IEEE Transactions on Power Systems, 38(4), pp.3514-3527.

[18] Rouzbahani, H.M., Karimipour, H. and Lei, L., 2020, October. An ensemble deep convolutional neural network model for electricity theft detection in smart grids. In 2020 IEEE international conference on systems, man, and cybernetics (SMC) (pp. 3637-3642). IEEE.

[19] Fida, K., Abbasi, U., Adnan, M., Iqbal, S. and Mohamed, S.E.G., 2024. A comprehensive survey on load forecasting hybrid models: Navigating the Futuristic demand response patterns through experts and intelligent systems. Results in Engineering, 23, p.102773.

[20] Khan, S., Mazhar, T., Shahzad, T., Ali, T., Ayaz, M., Ghadi, Y.Y., Aggoune, E.H.M. and Hamam, H., 2025. Optimizing load demand forecasting in educational buildings using quantum-inspired particle swarm optimization (QPSO) with recurrent neural networks (RNNs): a seasonal approach. Scientific Reports, 15(1), p.19349.

[21] Nandhini, N., Manikandan, V. and Elango, S., 2025. An interpretable generalized additive neural networks for electricity theft detection in smart cities using balanced data and intelligent grid management. Energy and Buildings, p.116123.

[22] Kumar, V.M., Bharatiraja, C., Elrashidi, A. and AboRas, K.M., 2024. Chaotic harris hawks optimization algorithm for electric vehicles charge scheduling. Energy reports, 11, pp.4379-4396.

[23] Cai, Q., Li, P., Zhao, Z. and Wang, R., 2024. Dynamic electricity theft behavior analysis based on active learning and incremental learning in new power systems. International Journal of Electrical Power & Energy Systems, 162, p.110309.

[24] Zhu, L., Wen, W., Li, J., Zhang, C., Zhou, B. and Shuai, Z., 2023. Deep active learning-enabled cost-effective electricity theft detection in smart grids. IEEE Transactions on Industrial Informatics, 20(1), pp.256-268.

[25] Khan, I.U., Javaid, N., Taylor, C.J. and Ma, X., 2022. Robust data driven analysis for electricity theft attack-resilient power grid. IEEE Transactions on Power Systems, 38(1), pp.537-548.

[26] Zhang, X., Liu, L., Lan, M., Song, G., Xiao, L. and Chen, J., 2022. Interpretable machine learning models for crime prediction. Computers, Environment and Urban Systems, 94, p.101789.

[27] Xia, X., Lin, J., Jia, Q., Wang, X., Ma, C., Cui, J. and Liang, W., 2023. ETD-ConvLSTM: A deep learning approach for electricity theft detection in smart grids. IEEE Transactions on Information Forensics and Security, 18, pp.2553-2568.

[28] Ding, H., Sun, Y., Huang, N., Shen, Z., Wang, Z., Iftekhar, A. and Cui, X., 2023. RVGAN-TL: A generative adversarial networks and transfer learning-based hybrid approach for imbalanced data classification. Information Sciences, 629, pp.184-203.

[29] Machine Learning on Imbalanced datasets - SBI Rostock. (n.d.). https://www.sbi.uni-rostock.de/research/projects/detail/59

[30] Bej, S., Davtyan, N., Wolfien, M., Nassar, M. and Wolkenhauer, O., 2021. LoRAS: An oversampling approach for imbalanced datasets. Machine Learning, 110(2), pp.279-301.

[31] Dagal, I., Tanriöven, K., Nayir, A. and Akın, B., 2025. Adaptive stochastic gradient descent (SGD) for erratic datasets. Future Generation Computer Systems, 166, p.107682.

[32] Gess, B., Gvalani, R.S. and Konarovskyi, V., 2025. Conservative SPDEs as fluctuating mean field limits of stochastic gradient descent. Probability Theory and Related Fields, pp.1-69.

[33] Kim, D.D., Chandra, R.S., Yang, L., Wu, J., Feng, X., Atalay, M., Bettegowda, C., Jones, C., Sair, H., Liao, W.H. and Zhu, C., 2024. Active learning in brain tumor segmentation with uncertainty sampling and annotation redundancy restriction. Journal of Imaging Informatics in Medicine, 37(5), pp.2099-2107.

[34] Dar, T.H. and Singh, S., 2025. Optimized parameter estimation of lithium-ion batteries using an improved cuckoo search algorithm under variable temperature profile. e-Prime-Advances in Electrical Engineering, Electronics and Energy, 11, p.100902.

[35] Mewada, H., Pires, I.M., Engineer, P. and Patel, A.V., 2024. Fabric surface defect classification and systematic analysis using a cuckoo search optimized deep residual network. Engineering Science and Technology, an International Journal, 53, p.101681.

[36] Ghori, K.M., Abbasi, R.A., Awais, M., Imran, M., Ullah, A. and Szathmary, L., 2019. Performance analysis of different types of machine learning classifiers for non-technical loss detection. IEEE Access, 8, pp.16033-16048.

[37] al-Ani, O., Das, S. and Wu, H., 2023. Imitation learning with deep attentive tabular neural networks for environmental prediction and control in smart home. Energies, 16(13), p.5091.

[38] Kaleem, W., Tewari, S., Fogat, M. and Martyushev, D.A., 2024. A hybrid machine learning approach based study of production forecasting and factors influencing the multiphase flow through surface chokes. Petroleum, 10(2), pp.354-371.

[39] Okoye, K. and Hosseini, S., 2024. T-test statistics in R: Independent samples, paired sample, and one sample T-tests. In R programming: Statistical data analysis in research (pp. 159-186). Singapore: Springer Nature Singapore.

[40] Shafin, S.S., 2024. An explainable feature selection framework for web phishing detection with machine learning. Data Science and Management.

[41] Ozdemir, G., Kuzlu, M., Sarp, S., Catak, F.O., Dimd, B.D. and Cali, U., 2024. The role of explainable artificial intelligence (xai) in smart grids. In Big data application in power systems (pp. 349-370). Elsevier Science.

[42] Noorchenarboo, M. and Grolinger, K., 2025. Explaining deep learning-based anomaly detection in energy consumption data by focusing on contextually relevant data. Energy and Buildings, 328, p.115177.