Feature Engineering for Semi-supervised Electricity Theft Detection in AMI

Elijah Orozco, Ruobin Qi, Jun Zheng
Department of Computer Science and Engineering
New Mexico Institute of Mining and Technology
Socorro, NM 87801
{elijah.orozco, ruobin.qi}@student.nmt.edu, jun.zheng@nmt.edu

2023 IEEE Green Technologies Conference (GreenTech) | 978-1-6654-9287-4/23/$31.00 ©2023 IEEE | DOI: 10.1109/GreenTech56823.2023.10173841

***Abstract*—Cyber attacks targeting Advanced Metering Infrastructure (AMI) of smart grids like electricity theft can cause significant financial losses to utility companies. Data-driven electricity theft detection methods based on machine learning techniques have become popular in recent years due to the massive amount of data and information collected by smart meters in AMI. Those methods are mainly based on unsupervised learning or supervised learning, both with limitations. In this paper, we proposed to use semi-supervised outlier detection for data-driven electricity theft detection, which only uses normal energy usage data to train detection models. Nine semi-supervised outlier detection algorithms were investigated in our study. To improve the performance of those algorithms, we performed feature engineering to extract 20 time-series features from energy load profiles, which help detect malicious changes in the load curve or energy usage amount. The performance of semi-supervised detection algorithms with and without feature engineering was evaluated with a real-world smart meter dataset under eight different False Data Injection (FDI) attacks. The results demonstrate that the proposed feature engineering offers a significant performance improvement for semi-supervised outlier detection algorithms.**

***Index Terms*—Advanced metering infrastructure (AMI), electricity theft detection, false data injection (FDI), semi-supervised outlier detection, feature engineering.**

I. Introduction

Modern smart grids utilize advanced metering infrastructure (AMI) to provide a bidirectional energy and information flow for efficient and resilient energy delivery and management [1]. A large number of smart meters are installed in the AMI network by utility companies to collect a massive amount of high-frequency energy consumption data from customers to provide data-driven energy delivery and management services. On the other hand, security vulnerabilities have also been introduced by AMI including the threat of electricity theft. Instead of physically destructing or bypassing mechanical metering to manipulate the readings, electricity thieves can launch cyber attacks to modify smart meter data to reduce electricity bills. Electricity theft has caused large amounts of financial losses to utility companies in different countries [1]. The timely detection of electricity theft is important for the security and sustainability of smart grids.

Traditionally, electricity theft detection relies on physical inspection or video surveillance which are time-consuming or costly [1]. By utilizing the massive amount of high-frequency smart meter data collected by AMI, data-driven electricity theft detection methods based on machine learning techniques have become popular in recent years which mainly use supervised learning or unsupervised learning. Supervised learning-based methods require labeled data of both normal and fraudulent users to train detection models. Shallow machine learning algorithms such as support vector machine (SVM) [2] and extreme gradient boosting (XGBoost) [3] have been applied for building detection models. Recently, the promising performance of deep learning in many fields has made it an ideal candidate for data-driven electricity theft detection. For example, Zheng et al. [4] developed a deep learning-based electricity theft detection method using a Wide & Deep Convolutional Neural Network (CNN) model which consists of two components: a wide component and a deep component. The wide component is a fully connected layer to learn the global knowledge from smart meter data while the deep component is a 2D CNN with multiple convolution layers, a pooling layer, and a fully connected layer to capture the periodicity of smart meter data.

Supervised learning-based methods require representative data samples of fraudulent users to build accurate detection models, which are difficult to obtain in real-world applications, if not impossible. On the other hand, unsupervised learning-based methods build models to identify potential fraudulent users only using unlabelled data. In [5], Zanetti et al. proposed an unsupervised pattern detection method for non-technical loss (NTL) detection, which compared fuzzy C-means (FCM), K-means, and Self-Organized Map (SOM) to determine the best-performed algorithm. A combined unsupervised method for electricity theft detection was proposed in [6] which integrates a correlation analysis using maximum information coefficient (MIC) and an advanced clustering technique, fast search and find of density peaks (CFSFDP). Qi et al. [1] proposed a novel unsupervised data-driven method for electricity theft detection in AMI which incorporates observer meter data, wavelet-based feature extraction, and FCM clustering. Unsupervised electricity theft detection methods have some limitations such as limited detection performance for certain attack types and relatively longer detection time.

Due to the limitations of supervised learning and unsupervised learning-based methods, we proposed to use semi-

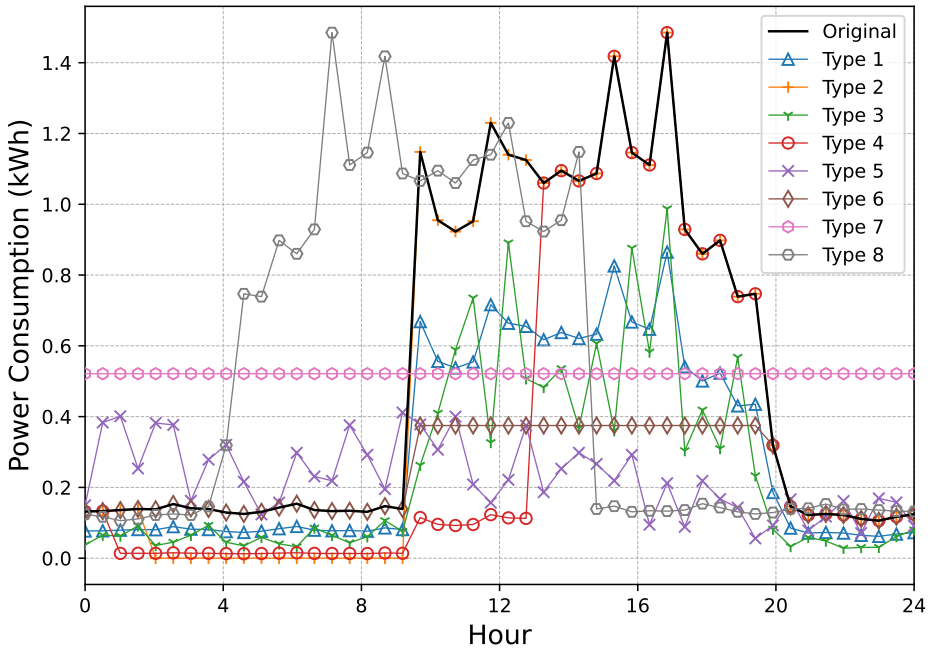


Fig. 1: A sample normal load profile and its corresponding tampered profiles by applying the eight types of FDI attacks

supervised outlier detection to solve the problem which only employs the data of normal users to train detection models. Nine different semi-supervised outlier detection algorithms were investigated in our study. Feature engineering was conducted to extract 20 time-series features from energy load profiles to improve the performance of semi-supervised outlier detection algorithms for electricity theft detection.

II. False Data Injection (FDI) Attacks

Since it is difficult for researchers to collect real-world smart meter data of electricity thieves, their malicious behaviors are modeled as various FDI attacks which modify normal load profiles to create malicious data for performance evaluation. Eight different FDI attacks shown in Table I were adopted in our study, which were widely used in other research works of this area [1]–[3], [6], [7]. In Table I, x is the normal load profile of a day. t is the time instance to collect a reading in a day. x_t is the reading of the normal load profile and $\tilde{x}_t$ is the modified reading of the tampered load profile at time t. p is the number of readings of a load profile. Fig. 1 shows a sample normal load profile and its corresponding tampered profiles by applying the eight types of FDI attacks.

As shown in Table I and Fig. 1, the type 1 attack reduces all readings by a fixed portion α to keep the same shape as the normal load profile where α is a random number between 0.2 and 0.8. The type 2 attack uses a random cutoff point, γ, to tamper with the normal load profile. Any reading of the normal load profile over γ is replaced by γ. For the type 3 attack, a random value γ less than the max reading of the normal load profile is subtracted from all readings of the profile. If the result is negative, the tampered value is set to 0. The type 4 attack randomly selects a time segment in a day that has a length of no less than 4 hours. All readings in the selected time segment are replaced with 0. The type 5 attack is similar to the type 1 attack except for the reduced portion α_t is randomly generated for each reading x_t. Thus, this attack generates a tampered load profile with a different shape than the normal one. The type 6 attack changes the load curve significantly by replacing the reading x_t with the average usage of the day, $\overline{x}$, multiplied by a randomly generated number α_t between 0.2 and 0.8. The type 7 attack replaces each reading x_t with $\overline{x}$ to keep the total energy usage the same as the normal load profile. The type 8 attack flips the order of all readings in a day. This attack changes the locations of the peaks and valleys of the profile while does not change the total usage. Both types 7 and 8 attacks try to subvert high electricity prices at specified time intervals set by utility companies without changing the total usage. An electricity theft can launch an attack with a mixture of the aforementioned attack types which is denoted

TABLE I: Eight types of FDI attacks

Attack Type	Modification
1	$\tilde{x}_t \leftarrow \alpha x_t,\ 0.2 < \alpha < 0.8$
2	$\tilde{x}_t \leftarrow \begin{cases} x_t, if x_t \le \gamma \\ \gamma, if x_t > \gamma \end{cases} \quad \gamma < max(x)$
3	$\tilde{x}_t \leftarrow max\{x_t - \gamma, 0\},\ \gamma < max(x)$
4	$\tilde{x}_t \leftarrow f(t) * x_t,\ f(t) = \begin{cases} 0, \text{ if } t_1 < t < t_2 \\ 1, otherwise \end{cases}$
5	$\tilde{x}_t \leftarrow \alpha_t x_t,\ 0.2 < \alpha_t < 0.8$
6	$\tilde{x}_t \leftarrow \alpha_t \overline{x},\ 0.2 < \alpha_t < 0.8$
7	$\tilde{x}_t \leftarrow \overline{x}$
8	$\tilde{x}_t \leftarrow x_{p-t}$

as the MIX type in this paper.

III. Methodology

A. Semi-supervised Outlier Detection Algorithms

Each of the nine semi-supervised outlier detection algorithms investigated in our study falls into one of the following categories: probabilistic, linear, proximity-based, ensemble, or neural network [8].

1) Angle-Based Outlier Detection (ABOD): ABOD is a probabilistic outlier detection method that is inspired by the idea that angles are more stable than distances [9]. If a data point is within a cluster of data points, the angle from it to a pair of other data points in the cluster varies widely. If the data point is an outlier, the angles tend to have a small variation. Thus, a threshold based on the variance of angles can be determined in the training stage for outlier detection.

2) Principal Component Analysis (PCA): PCA is a technique commonly used to reduce data dimensionality which can also be used to detect outliers [10]. A covariance matrix of the input, which contains features that are inter-correlated in general, is calculated to find eigenvectors where the first few of them preserve much of the data variance. When used for outlier detection, the outlier score of a data point is calculated as the sum of its projected distances on all eigenvectors.

3) One Class Support Vector Machine (OCSVM): OCSVM is an extension of the SVM algorithm for the one-class classification problem where only the data of the normal class is used for training [11]. The OCSVM algorithm maps input data into a high dimensional feature space and iteratively finds the maximal margin hyperplane that best separates the normal data from the origin, which is then used as the hyperplane to separate the normal and anomalous data [12].

4) Local Outlier Factor (LOF): LOF detects outliers by comparing a data point's reachability density to those of its neighbors [13]. A data point is designated as an outlier if its density is much smaller than those of its neighbors.

5) Cluster-Based Local Outlier Factor (CBLOF): CBLOF was developed in [14] to overcome the high-dimensionality issues suffered by LOF. LOF has to calculate the distances of every data point to its neighbors, resulting in a $O(n^2)$ computational complexity. CBLOF reduces the complexity to $O(n)$ by partitioning the data into clusters and only using the distance of a data point to its closest cluster. The outlier score of a data point is calculated based on a combination of the distance of a point to the closest cluster and the size of the cluster.

6) Histogram Based Outlier Detection (HBOS): HBOS is a fast outlier detection algorithm that calculates the outlier score by building a histogram for each input feature independently [15]. Since outliers tend to fall into the bins of low frequencies, the HBOS outlier score is calculated as the sum of the logarithmic uni-variate outlier scores of all input features where the uni-variate outlier score of a feature is defined as the inverse of the height of the bin the feature value of the data point falls into.

7) K-Nearest Neighbors (KNN): KNN is one of the simplest outlier detection algorithms, where the euclidean distance of a data point to its K-th nearest neighbor is used as the outlier score [16].

8) Isolation Forest (iForest): iForest uses an ensemble of binary search trees (BST) called isolation trees (iTrees) for outlier detection [17]. The outlier score of a data point is calculated based on the average path length obtained from all iTrees. Since an outlier tends to be isolated from other data points in the early partitioning of an iTree, it will have a short average path length which results in a large outlier score.

9) AutoEncoder: AutoEncoder is a multi-layer neural network that consists of three parts: encoder, code layer, and decoder. The encoder maps the input data into the code layer, representing the data in reduced dimensionality. The decoder then reconstructs the input data from the code layer. The training of AutoEncoder aims to minimize the difference between the input data and the reconstructed output. When used for outlier detection, the reconstruction error between the input data and the reconstructed output of the decoder is used as the outlier score [18].

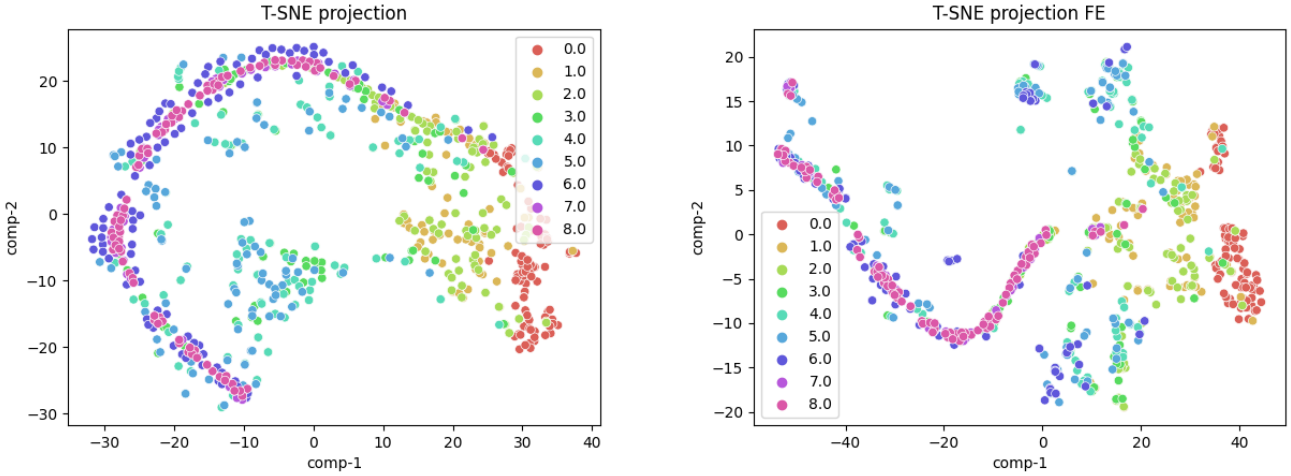
B. Feature Engineering

To improve the performance of semi-supervised outlier detection algorithms, we performed feature engineering which extracts 20 time-series features listed in Table II from a load profile. These 20 features were chosen to help detect malicious changes in the load curve or total energy usage amount. For example, $mean$, $standard\_deviation$, $variance$, $skewness$, $maximum$, $minimum$, and sum_values are popular features that give the statistical information on a user's daily usage behavior, which are useful for identifying whether the total usage or load curve stays similar to normal days or not. The $mean\_absolute\_change$ feature was chosen to help identify attacks such as types 2, 4, and 6 to 8, which cause significant changes in the load curve. This feature is similar to $standard\_deviation$ but it measures the relative distance between readings instead of measuring the distance from the mean, which makes it useful for differentiating the normal change of the load curve and the abnormal change after an attack. The $longest\_strike\_below\_mean$ and $longest\_strike\_above\_mean$ features can help detect the load curve changes by analyzing how long the readings stay below or above the mean usage. The energy usage of a normal day generally has a peak or peaks during the most active times of the day and consistent lows during inactive times. When the load curves are changed by attacks, the peaks may become shorter or the lows may become longer, which can be detected by the two features.

To show the effect of featuring engineering on differentiating normal load profiles and tampered load profiles, we used the t-Distributed Stochastic Neighbor Embedding (t-SNE) technique [19] which is a statistical dimension reduction method for visualizing high dimensional data by mapping high dimensional data as clusters in the 2-D plane. Fig. 2 shows two t-SNE graphs generated from the normal and tampered load

TABLE II: Twenty time-series features extracted from the load profile

Feature Name	Definition
mean	the mean value of the load profile
standard_deviation	the standard deviation of the load profile
variance	the variance of the load profile
benford_correlation	the correlation from first digit distribution when compared to the Newcomb-Benford's Law distribution
mean_abs_change	the mean over the absolute differences between subsequent readings of the load profile
skewness	the sample skewness of the load profile
longest_strike_below_mean	the length of the longest consecutive sub-sequence in the load profile that is less than the mean value
longest_strike_above_mean	the length of the longest consecutive sub-sequence in the load profile that is larger than the mean value
last_location_of_maximum	the relative last location of the maximum value of the load profile
first_location_of_maximum	the first location of the maximum value of the load profile
maximum	the highest reading of the load profile
minimum	the lowest reading of the load profile
has_duplicate_max	a Boolean value to check if the maximum reading of the load profile is observed more than once
has_duplicate	a Boolean value to check if any reading of the load profile occurs more than once
sum_values	the sum of all readings of the load profile
count_below_mean	the number of readings lower than the mean value
percentage_of_reoccurring_values_to_all_values	the percentage of non-unique readings of the load profile
sum_of_reoccurring_data_points	the sum of all readings that occurred more than once in the load profile
number_peaks	the number of peaks in the load profile



(a) (b)

Fig. 2: t-SNE graphs for normal and tampered load profiles, (a) without feature engineering, (b) with feature engineering (0.0: normal, 1.0 to 8.0: types 1 to 8 attacks)

profiles without and with feature engineering, respectively. It can be seen that with feature engineering, the clusters of tampered load profiles under different FDI attacks are denser and easier to separate from the cluster of normal load profiles. The result implies that our feature engineering has the potential to improve the detection performance of semi-supervised outlier detection algorithms.

IV. Performance Evaluation and Results

The dataset adopted for our performance evaluation is publicly available from the Irish CER Smart Meter Project [20]. The 180-day load profiles of 150 randomly selected small-medium-sized enterprises (SMEs) from July 15, 2009, to January 11, 2010, were used in our experiments. The readings of a load profile or the extracted features were normalized with min-max normalization before using as the input of a semi-supervised outlier detection algorithm. Five-fold cross-validation was adopted to test the performance of a semi-supervised outlier detection algorithm with or without feature engineering under one of the eight FDI attack types using an SME's 180-day load profiles. We tampered with half of the load profiles in the test fold using a chosen attack type which results in an outlier ratio of 0.5. We also performed the test under the MIX attack type where each attack type has an equal chance to be selected. The two metrics used for performance evaluation are the Area Under the Curve (AUC)

TABLE III: Average AUC values of semi-supervised outlier detection algorithms without feature engineering

FDI Attack Type	AutoEncoder	PCA	ABOD	HBOS	OCSVM	CBLOF	LOF	KNN	iForest
1	0.755	0.763	0.829	**0.833**	0.827	0.795	0.788	0.759	0.739
2	0.575	0.646	**0.766**	0.495	0.700	0.739	0.755	0.722	0.526
3	0.710	0.820	0.810	0.814	**0.879**	0.773	0.867	0.740	0.676
4	0.734	0.714	0.756	0.709	0.754	0.719	**0.791**	0.700	0.647
5	0.697	0.673	0.696	0.774	0.723	0.659	**0.796**	0.726	0.770
6	0.747	**0.834**	0.818	0.770	0.695	0.777	0.806	0.649	0.732
7	0.676	0.671	**0.776**	0.724	0.539	0.625	0.645	0.620	0.698
8	0.740	0.736	0.723	0.629	0.705	0.719	**0.761**	0.714	0.719
MIX	0.685	0.665	0.637	0.710	0.673	0.635	**0.732**	0.613	0.664

TABLE IV: Average F_1 scores of semi-supervised outlier detection algorithms without feature engineering

FDI Attack Type	AutoEncoder	PCA	ABOD	HBOS	OCSVM	CBLOF	LOF	KNN	iForest
1	0.697	0.716	0.731	0.697	0.718	0.712	**0.724**	0.709	0.704
2	0.570	0.644	**0.700**	0.566	0.651	0.688	0.687	0.682	0.555
3	0.763	**0.783**	0.744	0.696	0.789	0.721	0.734	0.722	0.694
4	0.748	0.742	0.724	0.696	**0.751**	0.697	0.744	0.687	0.699
5	0.728	0.716	0.696	0.717	0.715	0.680	**0.746**	0.643	0.721
6	0.744	**0.745**	0.686	0.715	0.689	0.665	0.744	0.64	0.726
7	0.680	0.680	0.585	0.678	0.579	0.622	0.661	0.620	**0.686**
8	0.725	0.717	0.688	0.656	0.700	0.673	**0.726**	0.678	0.709
MIX	0.704	0.697	0.638	0.684	0.689	0.651	**0.712**	0.626	0.698

TABLE V: Average AUC values of semi-supervised outlier detection algorithms with feature engineering

FDI Attack Type	AutoEncoder	PCA	ABOD	HBOS	OCSVM	CBLOF	LOF	KNN	iForest
1	0.809	**0.903**	0.893	0.834	0.857	0.863	0.886	0.862	0.779
2	0.664	**0.813**	0.666	0.722	0.597	0.613	0.640	0.621	0.617
3	0.903	0.951	0.944	0.928	**0.953**	0.920	0.926	0.922	0.917
4	**0.894**	0.884	0.815	0.859	0.683	0.794	0.793	0.801	0.772
5	**0.895**	0.892	0.878	0.859	0.806	0.848	0.846	0.855	0.828
6	**0.919**	0.916	0.902	0.883	0.798	0.880	0.861	0.892	0.904
7	0.874	0.872	0.842	0.820	0.773	0.843	0.822	0.850	**0.896**
8	0.807	**0.810**	0.759	0.801	0.656	0.745	0.736	0.755	0.794
MIX	**0.842**	0.835	0.828	0.820	0.761	0.815	0.808	0.802	0.802

TABLE VI: Average F_1 Score of semi-supervised outlier detection algorithms with feature engineering

FDI Attack Type	AutoEncoder	PCA	ABOD	HBOS	OCSVM	CBLOF	LOF	KNN	iForest
1	0.721	**0.854**	0.841	0.805	0.822	0.832	0.839	0.831	0.730
2	0.648	**0.794**	0.663	0.685	0.571	0.645	0.644	0.631	0.642
3	0.857	0.858	0.858	0.810	**0.889**	0.840	0.853	0.863	0.853
4	**0.855**	0.852	0.829	0.825	0.626	0.720	0.714	0.732	0.721
5	0.756	**0.856**	0.851	0.726	0.705	0.734	0.735	0.753	0.746
6	0.872	**0.874**	0.858	0.824	0.700	0.741	0.741	0.767	0.863
7	0.756	0.753	0.713	0.695	0.675	0.708	0.703	0.725	**0.854**
8	0.709	0.706	0.683	0.690	0.600	0.681	0.675	0.681	**0.711**
MIX	0.732	**0.733**	0.719	0.712	0.675	0.715	0.708	0.727	0.728

and F_1 score. The calculation of F_1 score needs to determine a detection threshold first. We adopted the approach of [21] which determines the threshold based on the outlier ratio of the test set. The results of the 150 SMEs were averaged to obtain the final result.

The results of the nine semi-supervised outlier detection algorithms without feature engineering in terms of average AUC value and F_1 score are presented in Tables III and IV, respectively, and the results of the algorithms with feature engineering are presented in Tables V and VI. For each FDI attack type, the best result is highlighted in bold. The results clearly demonstrate that our feature engineering significantly improves the performance of semi-supervised outlier detection algorithms under different FDI attacks. For example, under the attack of the MIX type, the performance improvements of the algorithms vary from 10.4% to 30.8% in terms of the average AUC value.

To have a fair performance comparison of different semi-supervised outlier detection algorithms with feature engineering across all attack types, we performed a Friedman test with a significance level of 0.05 [22]. The average ranks of the algorithms across all attack types based on the average AUC value were used in the test. Among all algorithms, PCA has the lowest average rank of 1.75 followed by AutoEncoder (3.5) and ABOD (3.625). The Friedman test indicated that there is a significant difference between the algorithms. Since the null hypothesis of the Friedman test was rejected, the post-hoc Nemenyi test was conducted for the pair-wise performance comparison, which showed that PCA significantly outperforms OCSVM, CBLOF, and LOF at the 95% confidence level,

and iForest at the 90% confidence level. Although there is no significant difference between PCA and the other four algorithms, it is clear that PCA combined with our feature engineering has great potential for semi-supervised electricity theft detection.

V. Conclusion

In this paper, we conducted feature engineering to improve the performance of semi-supervised outlier detection algorithms for electricity theft detection. 20 time-series features were extracted from load profiles to help the detection of malicious changes in the load curve or energy usage amount caused by FDI attacks. The performance evaluation results demonstrate that the proposed feature engineering can significantly improve the detection performance of semi-supervised outlier detection algorithms. In the future, automated feature selection to further improve detection performance will be studied.

Acknowledgment

This material is based upon work supported by the National Science Foundation EPSCoR Cooperative Agreement OIA-1757207.

References

[1] R. Qi, J. Zheng, Z. Luo, and Q. Li, "A novel unsupervised data-driven method for electricity theft detection in AMI using observer meters," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–10, 2022.

[2] P. Jokar, N. Arianpoo, and V. C. Leung, "Electricity theft detection in AMI using customers' consumption patterns," *IEEE Transactions on Smart Grid*, vol. 7, no. 1, pp. 216–226, 2015.

[3] Z. Yan and H. Wen, "Electricity theft detection base on extreme gradient boosting in AMI," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–9, 2021.

[4] Z. Zheng, Y. Yang, X. Niu, H.-N. Dai, and Y. Zhou, "Wide and deep convolutional neural networks for electricity-theft detection to secure smart grids," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 4, pp. 1606–1615, 2017.

[5] M. Zanetti, E. Jamhour, M. Pellenz, M. Penna, V. Zambenedetti, and I. Chueiri, "A tunable fraud detection system for advanced metering infrastructure using short-lived patterns," *IEEE Transactions on Smart Grid*, vol. 10, no. 1, pp. 830–840, 2017.

[6] K. Zheng, Q. Chen, Y. Wang, C. Kang, and Q. Xia, "A novel combined data-driven approach for electricity theft detection," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 3, pp. 1809–1819, 2018.

[7] M. G. Chuwa and F. Wang, "A review of non-technical loss attack models and detection methods in the smart grid," *Electric Power Systems Research*, vol. 199, p. 107415, 2021.

[8] Y. Zhao, Z. Nasrullah, and Z. Li, "Pyod: A python toolbox for scalable outlier detection," *Journal of Machine Learning Research*, vol. 20, no. 96, pp. 1–7, 2019.

[9] H.-P. Kriegel, M. Schubert, and A. Zimek, "Angle-based outlier detection in high-dimensional data," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2008, pp. 444–452.

[10] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley interdisciplinary reviews: computational statistics*, vol. 2, no. 4, pp. 433–459, 2010.

[11] B. Schölkopf, R. C. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt, "Support vector method for novelty detection," *Advances in Neural Information Processing Systems*, vol. 12, 1999.

[12] Y. Wang, J. Wong, and A. Miner, "Anomaly intrusion detection using one class SVM," in *Proceedings of the Fifth Annual IEEE SMC Information Assurance Workshop*, 2004, pp. 358–364.

[13] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, 2000, pp. 93–104.

[14] Z. He, X. Xu, and S. Deng, "Discovering cluster-based local outliers," *Pattern Recognition Letters*, vol. 24, no. 9-10, pp. 1641–1650, 2003.

[15] M. Goldstein and A. Dengel, "Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm," *KI-2012: poster and demo track*, vol. 9, 2012.

[16] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," in *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, 2000, pp. 427–438.

[17] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *Proceedings of 2008 Eighth IEEE International Conference on Data Mining*, 2008, pp. 413–422.

[18] C. C. Aggarwal *et al.*, *Data mining: the textbook*, 2015, vol. 1.

[19] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE." *Journal of Machine Learning Research*, vol. 9, no. 11, 2008.

[20] Commission for Energy Regulation, "CER Smart Metering Project—Electricity customer behaviour trial, 2009–2010," Irish Soc. Sci. Data Arch., Dublin, Ireland, SN: 0012-00, 2012. Available: https://www.ucd.ie/issda/data/commissionforenergyregulationcer/.

[21] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, "Deep autoencoding gaussian mixture model for unsupervised anomaly detection," in *International Conference on Learning Representations*, 2018.

[22] Y. Liu, Z. Li, C. Zhou, Y. Jiang, J. Sun, M. Wang, and X. He, "Generative adversarial active learning for unsupervised outlier detection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 8, pp. 1517–1528, 2019.