Network Intrusion Detection in Smart Grids for Imbalanced Attack Types Using Machine Learning Models

Dipanjan Das Roy and Dongwan Shin
Secure Computing Laboratory
Computer Science and Engineering
New Mexico Tech
Socorro, NM 87801 USA
{dipanjan.dasroy@student.nmt.edu, dongwan.shin@nmt.edu}

***Abstract*—Smart grid has evolved as the next generation power grid paradigm which enables the transfer of real time information between the utility company and the consumer via smart meter and advanced metering infrastructure (AMI). These information facilitate many services for both, such as automatic meter reading, demand side management, and time-of-use (TOU) pricing. However, there have been growing security and privacy concerns over smart grid systems, which are built with both smart and legacy information and operational technologies. Intrusion detection is a critical security service for smart grid systems, alerting the system operator for the presence of ongoing attacks. Hence, there has been lots of research conducted on intrusion detection in the past, especially anomaly-based intrusion detection. Problems emerge when common approaches of pattern recognition are used for imbalanced data which represent much more data instances belonging to normal behaviors than to attack ones, and these approaches cause low detection rates for minority classes. In this paper, we study various machine learning models to overcome this drawback by using CIC-IDS2018 dataset [1].**

***Index Terms*—smart grid, security, intrusion detection, imbalanced data, machine learning**

I. Introduction

The smart grid improves the efficiency, reliability and economics of current energy systems through the integrated use of both information technology (IT) systems used for data-centric computing and operational technology (OT) systems used to monitor events, processes and devices. Using the two-way flow of electricity and information, the smart grid builds an automated highly distributed energy delivery network, which supports real-time data exchange with the aim to balance supply and demand. The main services the smart grid provides include: automatic meter reading, power grid monitoring, demand side management, home networking between electrical devices for energy management.

However, there have been growing concerns over the security and privacy issues in smart grid systems, due partly to integrated communication networks among many devices based on both IT and OT systems. This network convergence creates a new domain of security and privacy issues ranging from catastrophic attacks to plenty of explanatory information available to an adversary, such as leakage of social security number, home address, lifestyle in terms of knowing when a customer is at home versus away from home, stealing of power, and attacks to disrupt network operations [2], [3]. Furthermore, because of strict latency requirements and the critical nature of power systems, the smart grid is also very susceptible to denial of service (DoS) attacks for which blackouts can happen.

Intrusion detection is a critical security service to protect smart grid systems, alerting the system operator for the presence of ongoing attacks. Hence, there has been lots of research conducted on intrusion detection, especially anomaly-based intrusion detection, to address security concerns in smart grids. However, some of those prior research is based on imbalanced data, which have much more data instances belonging to normal behaviors than to attack ones, and problems emerge when common approaches of pattern recognition are used for those imbalanced data; these approaches cause low detection rates for minority classes. In this paper, we investigate various machine learning models to overcome this drawback by using CIC-IDS2018 dataset [1].

II. Background and Related Work

A. Attack Scenarios

There could be lots of different types of attacks possible in smart grid systems. In order to better understand common situations and our proposed approach, let us first walk through some of attack examples in smart grids in detail.

Attack Scenario 1: A remote attacker exploits cross-site scripting (XSS) and SQL injection vulnerabilities to gain access to the vendor specific server and performs remote code execution on smart meters. In addition, once the attacker has access to the smart meter collector, he could launch the OpenSSL-based heartbleed to gain access to the smart grid server from which he targets the billing engine server.

Attack Scenario 2: A distributed DoS (DDoS) attack can deplete the computational resources of the target and cause serious delay or even failure in the data transmission or power service in smart grids. Hence, the impact of DDoS attacks

is the loss of availability of the smart grid. Unlike a DoS attack where the attacker makes the target unavailable to its legitimate users, a DDoS attack is a coordinated DoS attack where the attacker exploits multiple compromised systems such as botnets, to cause greater damage to the smart grid.

Attack Scenario 3: Malicious devices may be inadvertently infiltrated inside the trusted perimeter by personnel. For example, USB memory sticks have become a popular tool to circumvent perimeter defenses: if a few compromised USB sticks are plugged into previously secure devices inside the trusted perimeter, malware on the USB stick can immediately infect the devices.

Attack Scenario 4: A malicious user can access or change data in transit if a transmission application protocol for the data such as hypertext transfer protocol (HTTP) and file transfer protocol (FTP) is not protected. There are several approaches to transmission security: HTTPS using SSL (Secure Socket Layer) and TCP connections tunneling using well known SSH protocol. In supervisory control and data acquisition (SCADA) systems, to transfer files between host and remote sites for data logging, reporting or configuration, FTP is used.

All the above mentioned attacks are based on both IT and OT systems of the smart grid, and the CIC-IDS2018 dataset that we used for our research contains a handsome number of observations of those attacks.

B. Dataset for Intrusion Detection Research

Any mission critical or safety critical system connected to the Internet should have a strong security measure, especially a strong intrusion detection service, to protect itself from catastrophic failures, and smart grid systems are no exception. Machine learning has been playing a critical role in designing classification algorithms for intrusion detection systems, and KDD99 [14] has been the most widely used dataset for evaluating those algorithms. However, there are many problems associated with the KDD99 dataset. Two major problems are listed below:

- It is heavily imbalanced. Approximately 80% of data flow is attack traffic (3925650 attack instances in total 4898430 instances). Generally, a typical network contains approximately 99.99% of normal instances.
- Duplicate records in both training and testing datasets bias results for frequent DOS attacks and normal instances.

Duplicate and redundant records were removed from the KDD99 dataset and new dataset was named as NSL-KDD [15]. Since it is a re-sampled version of KDD99, some deficiencies still remain in NSL-KDD.

Canadian Institute of Cybersecurity (CIC) released a dataset called CIC-IDS2018 for intrusion detection research in 2018. The attacking infrastructure to create the dataset includes 50 machines and the victim organization has 5 departments with 420 machines including 30 servers. The dataset includes the captured network traffic and system logs of each machine, along with 81 features extracted from the captured traffic. The dataset includes 14 different attacks shown in Figure 1.

A	Benign	F	DDOS_attack_LOIC_UDP	K	DoS_attacks_Slowloris
B	Bot	G	DDoS_attacks_LOIC_HTTP	L	FTP_BruteForce
C	Brute_Force_Web	H	DoS_attacks_GoldenEye	M	Infilteration
D	Brute_Force_XSS	I	DoS_attacks_Hulk	N	SQL_Injection
E	DDOS_attack_HOIC	J	DoS_attacks_SlowHTTPTest	O	SSH_Bruteforce

Fig. 1. Attacks in CIC-IDS2018

A lot of attacks listed in Figure 1 have been observed in smart grid systems, as detailed in the previous section, and this is one of the reasons why we chose the dataset for our research.

C. Prior Intrusion Detection Research in Smart Grid

In general, networking devices in the smart grid have limited computation capabilities thus being very unlikely to have comprehensive security measures. Hence, utility companies need to calculate the risks of deployment and the necessity for intrusion detection before they choose systems that they want to monitor [6]. At the same time, intrusion detection algorithms are required to provide a precise decision on cyber attacks with minimal computational complexity.

Various intrusion detection approaches using machine learning were studied and evaluated based on KDD99 dataset in [7]. It was found that a higher degree of accuracy could be achieved by using those machine learning models. However, there were also some weaknesses that could result in misclassification of normal network data. Another research on intrusion detection was conducted for advanced metering infrastructure in [8]. The proposed state-based approach calculated security metrics using a sequence of events in the attack to achieve a high degree of confidence to develop intrusion detection for AMI. Similary, a two-tier intrusion detection framework was explored in [9] for AMI, and the approach proposed in citeb10 leveraged the concept of intrusion tolerance and was specifically designed to improve the availability of smart grid. Lastly, Yong Wang et al proposed an intrusion detection approach for SCADA systems to identify false data injection attacks [11].

Since the smart grid systems aim to provide resiliency against DDoS attacks, power outage and hardware/software failures, it is worthwhile to discuss some of the prior research focusing on the aspect. Leon Wu et al. proposed a reliability framework for the smart grid system, which evaluates the reliability of several stages and also provides necessary feedback for the betterment of the network safety in [12]. On the other hand, a probability-based model to analyze the result of intrusion detection and response on the reliability of the cyber-physical system was discussed in [13].

III. Our Approach

A. Intrusion Detection Classifiers Used

We have used four machine learning classifiers for our study, as follows:

a) Random Forest: Random Forest is a supervised learning algorithm. It consists of a collection or ensemble of simple tree predictors, each capable of producing a response when presented with a set of predictor values. For classification problems, this response takes the form of a class membership, which classifies a set of independent predictor values with one of the categories present in the dependent variable. This classifier organized a series of test questions and conditions in a tree structure. Once the tree has been constructed, classifying a test record is straightforward. Starting from the root node, apply the test condition to the record and follow the appropriate branch based on the outcome of the test. It then leads us either to another internal node, for which a new test condition is applied or to a leaf node. When the leaf node is reached, the class label associated with the leaf node is then assigned to the record it traces the path in the decision tree to predict the class label of the test record and the path terminates at a leaf node.

b) Naive Bayes: Naive Bayes is a simple probabilistic classifier that assumes the presence of a particular feature of a class which is unrelated to any other features. This means that the probability of one attribute does not affect the probability of the other. An advantage of Naive Bayes is that it only requires a small amount of training data to estimate the parameters (means and variances of the variables) necessary for classification. Because independent variables are assumed, only the variances of the variables for each class need to be determined and not the entire covariance matrix.

c) Extreme Gradient Boosting: XGBoost was mainly designed for speed and performance using gradient-boosted decision trees. XGBoost helps in exploiting every bit of memory and hardware resources for tree boosting algorithms. The algorithm is highly effective in reducing the computing time and provides optimal use of memory resources. XGBoost can perform the three major gradient boosting techniques, that is *Gradient Boosting*, *Regularized Boosting* and *Stochastic Boosting*. Boosting is a machine-learning algorithm and can be used to reduce bias and variance from the dataset. Boosting helps to convert weak learners to strong ones. A weak-learner classifier is weakly correlated with the true classification, whereas strong learners are strongly correlated. The main variation between various boosting algorithms is the method of weighting the training data and its hypothesis.

d) Support Vector Machine: Support vector machines (SVMs) are supervised learning models with associated learning algorithms that analyze data and recognize patterns used for classification and regression analysis. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on.

B. Data Preprocessing

Most real-world classification problems display some level of class imbalance, which is when each class does not make up an equal portion of the dataset. The data imbalance problem is recognized as one of the major problems in the field of data mining and machine learning as most machine learning algorithms assume that data is equally distributed. In the case of imbalanced data, majority classes dominate over minority classes, causing the machine learning classifiers to be more biased towards majority classes. This causes poor classification of minority classes. Classifiers may even predict all the test data as majority classes.

To overcome these challenges, several approaches have been developed that can be implemented during the pre-processing stage. One commonly used strategy is called resampling, which includes undersampling and oversampling techniques. If one balances the dataset by removing the instance from the overrepresented class then its called undersampling. Oversampling can be achieved by adding similar instances of underrepresented class to balance the skewed class ratio.

CIC-IDS2018 contains 15 different classes, 14 of them are attack types and one of them is benign. Out of a total of 16,233,002 (approx. 16 millions) observations we randomly selected 1,298,644 (approx. 8% of original dataset) observations while keeping the same proportionality of various attack types from the original dataset. A half of them (649,322) was kept aside as a testing set. From the remaining half we created a training set with 123,548 number of observations. The reason behind to shrink the training set is to balance the skewed class ratio. Following are the steps involved to generate the training set:

- Out of fifteen classes the following four classes have a very low number of observations:
  - Brute Force Web (611)
  - Brute Force XSS (230)
  - DDOS Attack LOIC UDP (1730)
  - SQL Injection (87)
- The minimum number of observations in the rest of the eleven classes is 10,903
- We apply combination of undersampling and oversampling in our training set.
  - As we have lots of observations of those eleven majority attack types we decided to use undersampling for those attack types.
  - For four minority classes we applied Synthetic Minority Over-sampling Technique (SMOTE) to make a balance dataset with the undersampled majority attack types.

1) Synthetic Minority Over-sampling Technique (SMOTE): Over-sampling by replication can lead to more specific regions in the feature space as the decision region for the minority class. This can potentially lead to overfitting on the multiple copies of minority class examples. To overcome the overfitting and broaden the decision region of the minority intrusion class cases, SMOTE can be used to generate synthetic examples

Label	Count
Benign	10921
Bot	10990
Brute_Force_Web	10990
Brute_Force_XSS	10990
DDOS_attack_HOIC	10990
DDOS_attack_LOIC_UDP	10990
DDoS_attacks_LOIC_HTTP	10901
DoS_attacks_GoldenEye	10990
DoS_attacks_Hulk	10990
DoS_attacks_SlowHTTP	10990
DoS_attacks_Slowloris	10990
FTP_BruteForce	10990
Infilteration	10903
SQL_Injection	10921
SSH_Bruteforce	10990

Label	Count
Benign	535660
Bot	11448
Brute_Force_Web	25
Brute_Force_XSS	9
DDOS_attack_HOIC	27441
DDOS_attack_LOIC_UDP	69
DDoS_attacks_LOIC_HTTP	23048
DoS_attacks_GoldenEye	1660
DoS_attacks_Hulk	18477
DoS_attacks_SlowHTTP	5596
DoS_attacks_Slowloris	440
FTP_BruteForce	7734
Infilteration	6431
SQL_Injection	4
SSH_Bruteforce	7504

Fig. 2. Training(left) and testing(right) set freq

by operating in feature space rather than in data space. The minority class is oversampled by taking each minority class sample and introducing synthetic examples along the line segments joining any of the k minority class nearest neighbors. Depending upon the amount of over-sampling required, neighbors from the k nearest neighbors are randomly chosen. Synthetic samples are generated in the following way: take the difference between the sample under consideration and its nearest neighbor, multiply this difference by a random number between 0 and 1, and add it to the feature vector under consideration. This causes the selection of a random point along the line segment between two specific features. This approach effectively forces the decision region of the minority class to become more general.

The synthetic examples cause the classifier to create larger and less specific decision regions, rather than smaller and more specific regions, as typically caused by over-sampling with replication. More general regions are now learned for the minority class rather than being subsumed by the majority class samples around them. The effect is that classifiers generalize better. This generalization capacity of a classifier can be very pertinent for intrusion detection. Figure 2 shows the frequency of training and testing set respectively.

C. Feature Selection

High dimensional data, in terms of a number of features, is common in smart grid networks. To extract useful information from these high volumes of data, we have to use statistical techniques to reduce the noise or redundant data. This is because we do not need to use every feature at our disposal to train a model. We can improve our model by feeding in only those features that are uncorrelated and non-redundant. This is where feature selection plays an important role. Not only it helps in training our model faster but also reduces the complexity of the model, makes it easier to interpret and improves the accuracy, precision and recall. We applied two feature selection algorithm (i) *Boruta Feature Selection* and (ii) *Recursive Feature Elimination* algorithms over CIC-IDS2018 dataset which are described below.

1) Boruta Feature Selection: The Boruta algorithm is a wrapper built around the random forest classification algorithm. It tries to capture all the important, interesting features in the dataset with respect to an outcome variable. It duplicates the dataset and shuffle the values in each column. These values are called shadow features. Then it trains a classifier, such as Random Forest Classifier on the dataset. By doing this we ensure that we have an idea of the importance for each of the features of our dataset. Then, the algorithm checks for each of our real features if they have higher importance. That is, whether the feature has a higher Z-score than the maximum Z-score of its shadow features. If they do, it records this in a vector and will continue with another iteration. In essence, the algorithm is trying to validate the importance of the feature by comparing with random shuffled copies which increases the robustness. Boruta can be very useful when we do not have any idea of the optimal number of features or we suppose that there are some features which are not contributing at all. Boruta discards fields which are not useful at all for the model. Therefore, we are removing features without subtracting from the model performance.

CIC-IDS2018 have total 81 features and after applying the Boruta feature selection algorithm on our training set we have 71 important features out of it. Since we still had a good number of features to work on, we decided to try one more feature selection algorithm on our training dataset.

1	Timestamp	14	Fwd.Pkt.Len.Max
2	Dst.Port	15	Fwd.IAT.Min
3	Init.Fwd.Win.Bytes	16	Fwd.IAT.Tot
4	Fwd.Seg.Size.Min	17	Fwd.Pkt.Len.Mean
5	Fwd.Header.Len	18	Tot.Fwd.Pkts
6	Init.Bwd.Win.Bytes	19	Subflow.Fwd.Pkts
7	TotLen.Fwd.Pkts	20	Fwd.Seg.Size.Avg
8	ACK.Flag.Cnt	21	Fwd.IAT.Max
9	Subflow.Fwd.Bytes	22	Flow.Pkts.s
10	RST.Flag.Cnt	23	Flow.Duration
11	ECE.Flag.Cnt	24	Fwd.IAT.Mean
12	Bwd.Pkt.Len.Std	25	URG.Flag.Cnt
13	Bwd.Header.Len	26	Pkt.Len.Std

Fig. 3. Selected Features from CIC-IDS2018

2) Recursive Feature Elimination (RFE): RFE is basically a backward selection of predictors. This technique begins by building a model on the entire set of predictors and computing an importance score for each predictor. The least important predictor are then removed, and the model is rebuilt and importance scores are computed again. The subset size that optimizes the performance criteria is used to select predictors based on the importance rankings. The optimal subset is then used to train the final model.

Not all models can be paired with the RFE method and some models benefit more from RFE than others. Because RFE requires that the initial model uses the full predictor set, some

models cannot be used when the number of predictors exceeds the number of samples. Backwards selection is frequently used with random forest and naive bayes models. We applied both of these models with RFE, random forest output with 26 most important features and naive bayes select 31 features as important. 26 features selected by random forest are also in common with naive bayes. We finalized these 26 features to work with different machine learning classifiers. The 26 features selected are shown in Figure 3.

	Precision	Recall	F1 Score
Benign	0.9999845	0.9640500	0.9816885
Bot	0.9957376	0.9999126	0.9978208
Brute_Force_Web	0.3333333	1.0000000	0.5000000
Brute_Force_XSS	0.6000000	1.0000000	0.7500000
DDOS_attack_HOIC	0.9950684	1.0000000	0.9975281
DDOS_attack_LOIC_UDP	1.0000000	1.0000000	1.0000000
DDoS_attacks_LOIC_HTTP	0.9907993	0.9998698	0.9953139
DoS_attacks_GoldenEye	0.9085933	1.0000000	0.9521078
DoS_attacks_Hulk	0.9998918	1.0000000	0.9999459
DoS_attacks_SlowHTTP	1.0000000	1.0000000	1.0000000
DoS_attacks_Slowloris	0.9341826	1.0000000	0.9659715
FTP_BruteForce	0.9996123	1.0000000	0.9998061
Infiltration	0.2568437	0.9993780	0.4086603
SQL_Injection	0.5714286	1.0000000	0.7272727
SSH_Bruteforce	0.9998667	0.9998667	0.9998667

Fig. 4. Precision, recall and F1 score of Random Forest

	Precision	Recall	F1 Score
Benign	0.9999067	0.9606691	0.9798952
Bot	0.9937456	0.9993012	0.9965156
Brute_Force_Web	0.1086956	1.0000000	0.1960784
Brute_Force_XSS	0.0006919	0.7777778	0.0013825
DDOS_attack_HOIC	1.0000000	0.9999636	0.9999817
DDOS_attack_LOIC_UDP	0.7204301	0.9710145	0.8271604
DDoS_attacks_LOIC_HTTP	0.9124920	0.9957914	0.9523236
DoS_attacks_GoldenEye	0.1584806	0.9777108	0.2727501
DoS_attacks_Hulk	0.9998310	0.9609785	0.9800198
DoS_attacks_SlowHTTP	0.8863201	0.9933881	0.9368048
DoS_attacks_Slowloris	0.5069124	1.0000000	0.6727828
FTP_BruteForce	0.9698853	0.9078097	0.9378214
Infiltration	0.9987486	0.9928471	0.9957891
SQL_Injection	0.0597014	1.0000000	0.1126760
SSH_Bruteforce	0.9972085	0.9997335	0.9984694

Fig. 5. Precision, recall and F1 score of Naive Bayes

IV. Performance Analysis

A. Performance Metrics

Accuracy alone is not a good evaluation option when working with an imbalanced dataset. As mentioned earlier,

	Precision	Recall	F1 Score
Benign	0.9036	0.9909	0.9452
Bot	0.9751	0.9998	0.9873
Brute_Force_Web	0.0439	0.5200	0.0809
Brute_Force_XSS	0.0000	0.0000	0.0000
DDOS_attack_HOIC	0.0000	0.0000	0.0000
DDOS_attack_LOIC_UDP	0.0000	0.0000	0.0000
DDoS_attacks_LOIC_HTTP	0.2011	0.2751	0.2323
DoS_attacks_GoldenEye	0.9908	0.3921	0.5619
DoS_attacks_Hulk	0.0000	0.0000	0.0000
DoS_attacks_SlowHTTP	0.0000	0.0000	0.0000
DoS_attacks_Slowloris	0.0974	0.1227	0.1086
FTP_BruteForce	0.5789	1.0000	0.7333
Infiltration	0.0000	0.0000	0.0000
SQL_Injection	0.0000	0.0000	0.0000
SSH_Bruteforce	0.0000	0.0000	0.0000

Fig. 6. Precision, recall and F1 score of SVM

	Precision	Recall	F1 Score
Benign	1.00000	0.98180	0.99080
Bot	0.99756	0.99991	0.99873
Brute_Force_Web	0.27780	1.00000	0.43480
Brute_Force_XSS	0.36000	1.00000	0.52940
DDOS_attack_HOIC	0.99528	1.00000	0.99764
DDOS_attack_LOIC_UDP	1.00000	1.00000	1.00000
DDoS_attacks_LOIC_HTTP	0.99097	1.00000	0.99546
DoS_attacks_GoldenEye	0.85084	1.00000	0.91941
DoS_attacks_Hulk	1.00000	1.00000	1.00000
DoS_attacks_SlowHTTP	1.00000	1.00000	1.00000
DoS_attacks_Slowloris	0.19298	1.00000	0.32352
FTP_BruteForce	0.99819	1.00000	0.99910
Infiltration	0.47265	1.00000	0.64191
SQL_Injection	0.80000	1.00000	0.88890
SSH_Bruteforce	0.99960	0.99987	0.99973

Fig. 7. Precision, recall and F1 score of XGBoost

	Random Forest	Naive Bayes	SVM	XGBoost
Precision	0.8390	0.6875	0.2527	0.7957
Recall	0.9975	0.9691	0.2867	0.9987
F1 Score	0.8850	0.7240	0.2433	0.8479

Fig. 8. Mean precision, recall and F1 score

	Random Forest	Naive Bayes	SVM	XGBoost
Accuracy	97.01	96.47	86.28	98.49

Fig. 9. Accuracy of machine learning models

dealing with a highly imbalanced dataset is one of the biggest challenges in IDS. An imbalance occurs when one or more classes have very low proportions in the data compared to the other classes. Most of the classification algorithms do not work well for such problems since the classifiers tend to be biased towards the majority class and hence perform poorly on the minority class. As a result, the high accuracy obtained from these classifiers is largely dominated by the majority classes. Hence, in addition to accuracy, we used the following metrics to evaluate our models:

a) Confusion Matrix: A clean and unambiguous way to present the prediction results of a classifier is to use a confusion matrix. This is a useful table that presents both the class distribution in the data and the classifiers predicted class distribution with a breakdown of error types.

b) Precision: Precision is the number of true positives divided by the sum of true positives and false positives. Put another way, it is the number of positive predictions divided by the total number of positive class values predicted. Precision can be thought of as a measure of a classifiers exactness. A low precision can also indicate a large number of false positives.

c) Recall: Recall is the number of true positives divided by the sum of true positives and false negatives. Put another way it is the number of positive predictions divided by the number of positive class values in the test data. Recall can be thought of as a measure of a classifiers completeness. A low recall indicates many false negatives.

d) F1 Score:: The F1 Score is the harmonic mean of precision and recall. Put another way, the F1 score conveys the balance between the precision and recall. If we want to create a balanced classification model with the optimal balance of recall and precision, then we try to maximize the F1 score.

B. Results

This study aimed at developing an efficient intrusion detection method with low false negative rates. For this purpose, Boruta Feature Selection and Recursive Feature Elimination was performed to determine a subset of the original features in order to eliminate the irrelevant features as well as to improve the classification efficiency. In classification step, Random Forest (RF), Naive Bayes (NB), Support Vector Machine (SVM) and Extreme Gradient Boosting (XGBoost) were trained and tested based on CIC-IDS2018 dataset. Out of these four classifiers, Extreme Gradient Boosting and Random Forest performed better with respect to precision, recall and F1 score. The main objective of this experiment is to acquire overall better classification model in terms of low false negative rates. The XGBoost and RF classifier achieved recall 99.87% and 99.75% respectively. These experiments were performed by R version 3.5.3 on desktop PC with 3.6 GHz Intel Core i7-4790 processor and 16GB RAM.

V. Conclusion

Though the classifiers used in this study have exhibited satisfactory performance there is still much room for improving their performance. The current classifier models will be improved through the inclusion of network packets and attacks that are unique to the smart grid. The parameters of the classifier models will be re-evaluated for a better training of the newly constructed data. It is also possible that some other classifiers may be able to achieve better performance. Thus, our immediate future work will focus on evaluating the effectiveness of some existing classifiers for our problem. In this way, for each attack type the most effective classifier can be determined.

Acknowledgment

This work was partially supported at the Secure Computing Laboratory at New Mexico Tech by the grant from National Science Foundation (NSF-OIA-1757207).

References

[1] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," 4th International Conference on Information Systems Security and Privacy (ICISSP), Portugal, January 2018.

[2] W. Wang, Y. Xu, and M. Khanna, " A survey on the communication architectures in smart grid," Computer Networks, 55(15):36043629, 2011.

[3] Y. Sun, X. Guan, T. Liu, and Y. Liu, "A cyber-physical monitoring system for attack detection in smart grid," in 2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), April 2013, pp. 3334.

[4] D. Evans, A. Nguyen-Tuong, and J. Knight, "Effectiveness of moving target defenses," Effectiveness of Moving Target Defenses, vol. 54, p. 2948, Aug 2011.

[5] M. Harvey, D. Long, and K. Reinhard, "Visualizing nistir 7628, guidelines for smart grid cyber security," in 2014 Power and Energy Conference at Illinois (PECI), Feb 2014, pp. 18.

[6] D. Grochocki, J. H. Huh, R. Berthier, R. Bobba, W. H. Sanders, A. A. Cierdenas, and J. G. Jetcheva, "Ami threats, intrusion detection requirements and deployment recommendations," in 2012 IEEE Third International Conference on Smart Grid Communications (SmartGridComm), Nov 2012, pp. 395400.

[7] R.K. Sharma, H.K.Kalita, P.Borah, "Analysis of Machine Learning Techniques Based Intrusion Detection Systems," Proceedings of the 3rd International Conference on Advanced Computing, Networking and Informatics, 2015, pp. 485-493.

[8] M.Rausch, B.Feddersen, K.Keefe, and W.H.Sanders, "A Comparison of Different Intrusion Detection Approaches in an Advanced Metering Infrastructure Network Using ADVISE," International Conference on Quantitative Evaluation of Systems, 2016, pp. 279-294.

[9] V.Gulisano, M.Almgren, and M.Papatriantafilou, "METIS: a Two-Tier Intrusion Detection System for Advanced Metering Infrastructures," International Conference on Security and Privacy in Communication Systems, 2015, pp. 51-68.

[10] M.Tanha, and F.Hashim, "Towards a Secure and Available Smart Grid Using Intrusion Tolerance," International Conference on Internet and Distributed Computing Systems, 2012, pp. 188-201.

[11] Y.Wang, Z.Xu, J.Zhang, L.Xu, H.Wang, and G.Gu, "SRID: State Relation Based Intrusion Detection for False Data Injection Attacks in SCADA," European Symposium on Research in Computer Security, 2014, pp. 401-418.

[12] L.Wu, and G.Kaiser, "Evaluating Machine Learning for Improving Power Grid Reliability," Workshop on Machine Learning for Global Challenges, Bellevue, WA, USA, 2011.

[13] R.Mitchell and I.R.Chen, "Effect of Intrusion Detection and Response on Reliability of Cyber Physical Systems," IEEE Transactions on Reliability, vol. 62, no. 1, 2013, pp 199 210.

[14] Kendall K, "A database of computer attacks for the evaluation of intrusion detection systems," Master's thesis, MIT - Massachusetts Instituteof Technology, 1999.

[15] M.Tavallaee, E. Bagheri, W.Lu, A.A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set," IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009.