Cyber-Security Audit for Smart Grid Networks: An Optimized Detection Technique Based on Bayesian Deep Learning

Alexander N. Ndife[1], Yodthong Mensin[1*], Wattanapong Rakwichian[1], and Paisarn Muneesawang[2]

[1]School of Renewable Energy and Smart Grid Technology
Naresuan University, Phitsanulok 65000 Thailand
alexandern60@email.nu.ac.th, yodthongm@nu.ac.th, wattanapong.r@gmail.com

[2]Department of Electrical and Computer Engineering
Naresuan University, Phitsanulok 65000 Thailand
paisarnmu@nu.ac.th

Received: October 24, 2021; Accepted: May 1, 2022; Published: May 31, 2022

Abstract

Security of computers, networks and their communication protocols are vital in smart grid technology operation and its management. This paper discusses a proposed Bayesian Neural Networks for time-series TCP/IP packets intrusion detection and threats classification in a grid network. This architecture termed SGtechNet detects invariants with maximized detection accuracy by applying a robust method that approximates the variation in posterior weights of neural networks with variational inference to minimize the divergence between prior and true network posterior distributions. Spatiotemporal feature engineering and uncertainty estimation in Bayesian modeling, were leveraged to learn novel attack features and classify attacks accordingly. This architecture reduced the size of the proposed model to 25 % of the size of a pioneer model (AlexNet), hence, facilitating the inference time compared to the baseline. SGtechNet was tested on NSL-KDD datasets using two deep learning algorithms: CNN-LSTM and GRU, on two classification categories (binary and multiclass) with Accuracy, Precision, Recall, and F1-Score as the performance metrics. GRU algorithm comparatively performed moderately well on both classification categories, unlike CNN-LSTM that performed convincingly only on one test category. Comparing the result of the SGtechNet model against a comparator model showed outstanding performance in both model size, computational speed, and marginal improvement in terms of accuracy. Chi-Square Test analysis determined that the degree at which the training accuracy differed with validation accuracy was statistically insignificant.

Keywords: Bayesian, Cyber Threat, Classification, Deep Learning, Intrusion detection, Neural Networks

1 Introduction

Security of smart grid is of great concern to researchers due to the importance of ensuring the reliability and stability of power system operations. Cybersecurity threats are the most confronting challenge facing smart grid networks with the business and economic disruption that occurs, and the potentially huge economic losses likely. Technological advancement in energy sector has given Integrated circuits

Journal of Internet Services and Information Security (JISIS), volume: 12, number: 2 (February), pp. 95-114
DOI:*10.22667/JISIS.2022.05.31.095*

*Corresponding author: School of Renewable Energy and Smart Grid Technology, Naresuan University 65000 Thailand, Tel: +66-893-59-1465

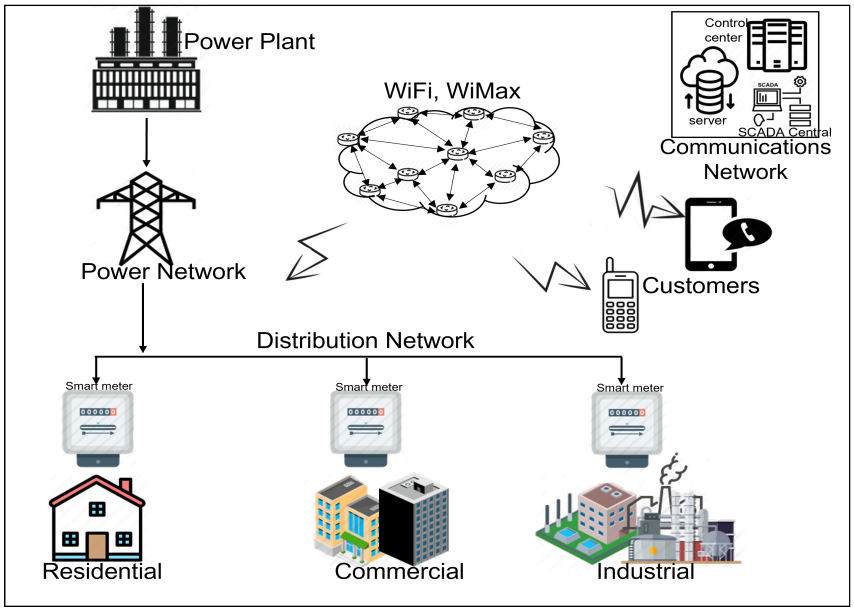

Figure 1: Communication Architecture of a Smart Grid

(ICs) and Supervisory Control and Data Acquisition (SCADA) key roles in smart grid infrastructure. Unfortunately, this has unintentionally exposed the power plant infrastructures such as advanced metering infrastructure (AMI) which encompasses smart meters, communication networks and data management systems [1] and sensors, to sophisticated cyber-attacks resulting to operational failure and loss of synchronization in the grid. There have been incidences in the recent past of attacks targeting industrial power plant like that of Siemens in 2010, Ukrainian power grid in 2015 by Russian hacker groups and many others. These attacks have caused blackouts, destruction of sensitive equipment, and an increase in energy costs and prices for access to the grids. It has been predicted that cybercrime damages will cost the world $6 trillion annually by 2021 and that a business will fall victim of a ransomware attack every 14 seconds, from the current 40 seconds, somewhere in the world [2].

The historical background of cyber threats in the world can be traced back to the “Morris worm”, of 1988 [3]which ravaged America’s cyber space. Subsequently, various examples of malicious code have emerged, disrupting computer programs, attacking the integrity and authenticity of data, resulting in the theft or even destruction of sensitive data. In 2007, Stuxnet malware was used to attack the Iran nuclear power station and was subsequently used in 2010 to exploit the vulnerabilities in the SCADA system’s power-grid operation forcing the control center to malfunction. These attacks are possible only when Internet’s Transmission Control Protocol/Internet Protocol (TCP/IP) networking environment of figure 1 and its security authentication process shown in figure 2 is flawed. Lack of intrusion-resistance in the network has given leeway for possible compromise using different malicious codes during intrusions by bad actors. SCADAs high dependence on open connectivity to corporate networks and the Internet, for example, makes it particularly vulnerable to attacks. Ironically, software-defined network-enabled smart grid emergence seems to have increased the level of vulnerability of power systems to cyber-attacks. Based on this fact, the compromising of sensitive smart grid network data has become the subject of increasing concern, necessitating a measure to detect potential intrusions and protect critical power system infrastructures.

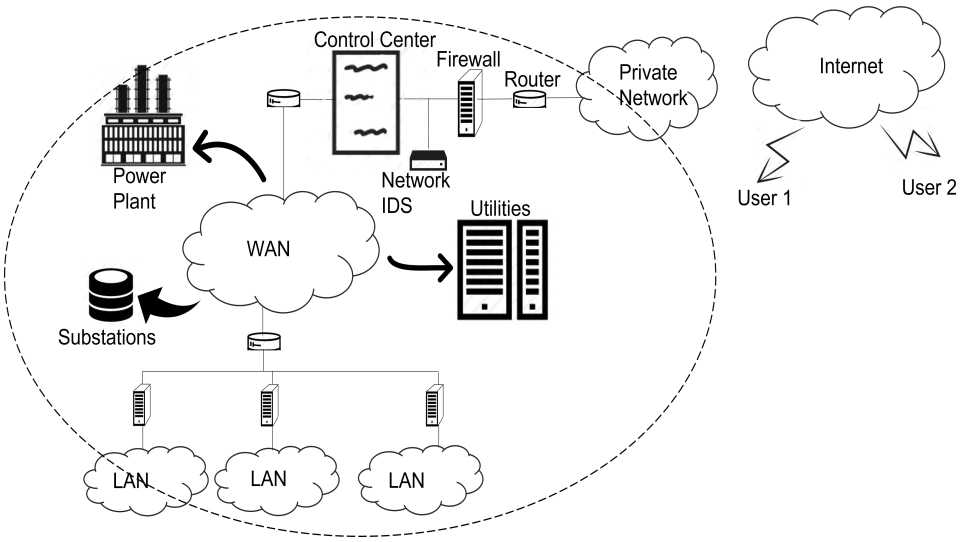

Figure 2: Defensive Mechanism of a Smart Grid Network

One of the most effective preventive measures against the risks of cyberspace breach is the thorough examination, or security risk audit, of the smart grid technologies before massive deployment [4]. In this context, the SGtechNet model that is presented in this paper is focused particularly on smart grids network given the greater volume of periodic traffic that they carry and their complex nature. Previous research efforts paid little or no attention to understanding the nature of these threats, especially the novel ones [5]. However, no defensive arrangement can guarantee protection without prior knowledge of existing attack patterns. Therefore, a supervised Bayesian deep learning architecture is used for proper analysis of grid network attack patterns to reliably estimate possible uncertainties. A time-frequency features-representation analysis carried out using SGtechNet captured the dominant spatiotemporal features in each attack types and approximates their prediction distributions over networks weights. Some researchers have recently used various methods of feature extraction and have come up with successful results. Improved model accuracy was achieved in [6] [7] [8] using extensive feature engineering technology that expressed the semantic and contextual information in a data.

The SGtechNet modeled the grid network traffic and classify novel attacks and non-attacks accordingly. Looking specifically at the Software-Defined Network (SDN) architectures of smart grids (comprising SCADA, micro grids, substations), some causes of vulnerability, such as single point failure in the SDN controller, or the inability to monitor and control flow dynamics, can be easily noticed. These loopholes cause congestion that results in Denial-of-Service (DoS) attacks and the inability to detect and respond to the attack. Large numbers of packets without matching rules in an OpenFlow network cause saturation in the communication between a switch and the controller, resulting in a DoS attack. However, applications in the security domain, that are based on artificial intelligence, have demonstrated that they can surpass human intelligence in relation to classification, recognition and prediction tasks [9]. These intelligent applications have also shown robustness in utilizing the knowledge learnt from the analysis of historical data to perform complex computational tasks, enabling human experts to make informed decisions. Deep Learning algorithms normally require a larger amount of data while machine learning requires less data [10]. Larger data samples also enhance the model performance. Deep learning has therefore become popular in security tasks such as malware detection, intrusion and spam detection, and general network surveillance [11]. This popularity can be linked to its ability to use densely interconnected networks of neurons to perform specialized tasks [9]. Combining Bayesian method and neural network architecture for intrusion detection in smart grid networks has a greater advantage. This can

enable the determination of statistical properties such as standard deviation and variance, so as to quantify the behavioral characteristics [12]. This is very important if we must ascertain what 'abnormality' is, and how it differs from 'normal' data content at any given time. Importantly, our proposed model is necessary due to the complex nature of often-hostile environments, such as SCADA networks where numerous real-time devices are connected over a long period of time [10]. With this type of heavy data traffic, the network and computer infrastructures are prone to incessant attacks by cyber criminals. Major contributions of this paper include modeling of smart grid networks attacks patterns in a real-time, optimization of supervised learning techniques used in predictive modeling using labeled training data [13] and provision of defensive mechanism against malware and other malicious objects that manipulate the operations of a grid network especially the cyber-physical systems.

2 Related Work

As a result of rising concerns about malicious attacks on the smart grid cyberspace, researchers have delved into finding measures to minimize these damaging trends. [14] x-rayed deep learning-based intrusion detection schemes in classifying malicious behaviours and categorize them accordingly. In order to ameliorate the proliferation of cyber security attacks, a problem-solving approach, described in [15], was created as the first vulnerability-assessment program to verify security settings through the Internet. Over time, this challenge has led to the deployment of different machine learning algorithms applying supervised, semi-supervised and unsupervised learning methods, to learn the attack patterns. These have variously used, *inter alia*, logistic regression, SVM, clustering, and Boltzmann machines, and deep neural networks. However, as continual efforts are being made to curtail possible malicious attacks, the dynamics of the attacks keep changing over time, making the task very tedious for human cyber security experts. Most statistical learning models in recent times are flawed when applied to adversarial sample-based attacks, especially stealthy ones, due to its operational dynamism. Individuals with malicious intent can easily circumvent deep learning-powered malware detection by exploiting feature amplitude disequilibrium, but [5] was able to address this particular shortcoming using randomly nullifying technique to reduce the effectiveness of the adversarial samples.

Recently, research efforts combining both statistical and neural networks methods like [16] were made to identify possible DOS and DDOS attacks. [16] determined the difference between the metric candidates and the attack labels in an IoT traffic data with statistical tools. A case study of attacks, in 2015, on a power station in Ivano-Frankirsk in Ukraine, where "BlackEnergy" was used to delete data and destroy hard drives, but also asserted a DOS attack on the support phone numbers of the power station managers, thereby preventing customers from notifying the utility company of the failure. This justified the proposed Bayesian neural network method which seeks to contain the flexibility in DOS attacks on the network devices which may include the theft of digital certificates and prevention of computers and servers from booting. A report by the security firm, Kaspersky Labs shows that even customized extensions and plugins like those seen in Microsoft and Linux can be used in carrying out DoS attacks, stealing passwords, scanning ports, logging IP sources, covertly taking screenshots, gaining persistent access to command-and-control channels, and destroying hard drives.

2.1 Bayesian Neural Networks

This paper leverage on Bayesian neural networks known for estimating predictive uncertainty and probabilistic interpretation of deep learning models' distributions over weights. However, the idea of modelling a distribution over weight, has been a challenge because of the runtime complexity of extremely large models [17] commonly used in practical applications that incurs a significant cost, necessitating ap-

proximations to the model posterior with variational inference [18]. This approximation therefore makes Bayesian predictive distribution which contains all the uncertainty to be tractable. Stochastic random process variables (with uncertainty at an instance of time), and discrete variables (features or attributes) in attacks on a grid network can be assumed to be conditionally independent of the discrete-valued variable C referred to as a class, which represents attack types. So, probability of an attack at an instant of time could be determined with the knowledge of the stochastic variational inference of posteriors. Gaussian process posterior computes acquisition function, $\alpha(x)$ of the random variables of x to decide where to estimate f function next. Point estimation of a function often results in extrapolations with unjustified high confidence in classification of uncertainty unlike distribution estimation. IP/TCP packet for example is considered here since the concern was to confirm if the packet is an attack or not, as in case of [19] where classification was based on binary class variables C. [20] proposed an entropy search method that minimizes information about the class variables, and [21] addressed the computational difficulties by rewriting the entropy search acquisition function with symmetry of mutual information into predictive entropy search. Similarly, to the case of multinomial classifications, where different types of malicious attacks are classified, we can assume the binary conditionality to be random variables (x,y) where each takes different numbers of discrete values, and features variables $x_0 \ldots x_p$ which are assumed to be the different network connection states, such as duration of the connection or connection attempts rejected etc.

The joint probability that $X = x$; $Y = y$, can be denoted by $p(x,y)$, expressed by the product rule in Equation 1:

$$p(x,y) = p\left(\frac{y}{x}\right)P(x) \tag{1}$$

where $P(x)$ is the prior probability of $X = x$, irrespective of value Y, $p(y/x)$ is the conditional probability of $Y = y$, given that $X = x$. This can be regarded as a function of C, called likelihood, expressed by the product rule in Equation 2:

$$p\left(\frac{y}{x}\right) = \frac{P(y/x)P(x)}{P(y)} \tag{2}$$

where $p(x/y)$ is the posterior probability which plays a central role in pattern recognition. Applying the sum rule, we have: $p(y) = p(y/x)$, which is a normalizing factor that ensures that the probability on the left-hand side of Equation 2 sums to one (in Equation 1), satisfying the probability function that a variable must take one of the values at an instant time. Assuming a neural network model $f^w(x)$ consists of sets of interconnected weights, w, predicting y from x can easily be achieved by optimizing the weights through gradient descent and back propagation using loss function. In this network, weights are modeled as distributions and prior distributions placed over them so that the posterior distribution of the weights can be found with Bayesian Inference. Uncertainty is modelled using dropout sampling created and ran multiple times at test time to create a distribution of outcomes. Predictive entropy for the classes in categories of classification is therefore calculated to determine the prediction probability for each class. The prior weights of the model were trained on standard categorical entropy loss and the loss function was continuously adjusted until an improved prediction accuracy is achieved.

2.2 Deep Learning Approach to Network Intrusion

Defensive mechanisms can be implemented in deep learning either as network-based [17] or host-based [22](depending on the choice and availability of resources, implemented using either signature-based, such as Snort [23], or anomaly-based techniques such as Bro [24]. Signature-based intrusion detection

techniques detect attacks by using known signatures or exploits the patterns of attacks on a knowledge-based description platform. Cases where incoming traffic payloads are compared with stored signatures, through the process of signature matching, has been very effective in detecting known threats but are almost ineffective at detecting previously unknown threats. Nonetheless, signature-based detection is widely implemented because it generates lower false alarm rates than anomaly-based techniques even though signature matching processing time is a drawback. Delays in the signature matching process led to overload packets, high consumption of computer resources, and false alarms. To address this, [25] used an exclusive signature matching scheme to identify the mismatch in a signature rather than an accurate match for the regular signature matching. [12] detect attacks in the network's payloads by comparing the similarity of the payload's frequency distribution and standard deviation, and [26] reduced high dimensional space features using feature selection method called Particle Swarm Optimization (PSO). On the other hand, anomaly-based detection uses well-defined rules as a template for normal behavior, anything outside those predefined rules is regarded as abnormal. Anomaly-based detection identifies an intrusion by detecting deviations between current events and predefined normal profiles, which means that there is a need to differentiate between events. This is a very difficult situation to model because of the nature of network traffic, which is flexible and irregular. Machine learning methods are usually challenged whenever complex data classification problems, such as data domain identification, and classification of class types, are encountered. In order to mitigate memory requirement drawbacks that this type of task is known for, [27] used Auto-encoder to learn the semantic similarities of the different features embedded in latent representations. This helped to reduce the dimensionality of the features learned.

Deep learning is an alternative to machine learning when addressing complex problems, having proved its worth in image classification and recognition tasks involving more than conventional sequential data processing [17]. Various deep learning optimization techniques have been introduced, including dropout for model regularization. This dropout technique can be likened to the Bayesian neural network used in this study which placed a probability distribution of weights over convolution kernels in the architecture for robustness in addressing model overfitting. [28] presented the first regularization method that uses the stochastic model averaging technique to improve the performance of deep learning models and tends to make classification less reliant on arbitrary units. [29] deployed a hybrid algorithm: k-means and fuzzy rules for anomaly-based detection. [30][31] used the LSTM-RNN classifier, and [32] modified the functionality of GRU using SVM as a classifier instead of conventional Softmax, for computational efficiency. Similarly, [33] modelled five deep learning classifiers for intrusion detection and performed comparative analysis on the five classifiers. Some research efforts have applied a slightly different approach, such as the Auto-encoder model that was applied in [27] to learn the latent representation similarities in features to detect abnormalities in the network. However, the problem of intensive computational time has not been adequately addressed in the literature until the recent use of the Bayesian method which allows values of regularization coefficients to be selected only in training thereby expediting the whole classification process. In our research, we leveraged a lightweight adaptive architecture, and some other methods reported in the literature, to improve model performance. Our major targets were to reduce over fitting and the computational complexity to hasten the detection process as well as to improve validation accuracy.

2.3 Uncertainty Estimation

To estimate the level of uncertainty in the predictions, the neural network approach was used. Uncertainties emanating in predictions of problems of this nature can either be because of noise in the dataset or inadequacies in modeling (misspecification) or model uncertainty. For the purposes of this study, modeling of uncertainty was based on inadequacies in modeling (epistemic), where weights w is modeled as distributions. Instead of initializing our weights to perform optimization to get the final weights, a

prior distribution was placed over the neural network weights (i.e., Gaussian prior), $p(w)$ to compute the posterior distribution to be used in quantifying the predictive uncertainty [34]. As in [35], the idea is to model the posterior with variational inference distribution and fit such closely distribution's parameters to the true posterior by minimizing the Kullback-Leibler (KL) divergence from variational distributions to the true posterior. [36] used approximation of model's entropy to decide the data points that minimize uncertainty about the model's parameters while [20] proposed Entropy search that maximizes the information search for class variables given a new data. Bayesian inference to find the posterior distribution of the weights $p(w|x,y)$from Bayes theorem is shown below.

The posterior distribution of W can be computed as:

$$p(w/x,y) = \frac{p(y/x,w)p(w)}{p(y/x)} = \frac{p(y/x,w)P(w)}{\int p\left(\frac{y}{x},w\right)dw} \tag{3}$$

where $p(y/x,w)$ is the likelihood.

Therefore, to calculate $p(y|x)$ we averaged this likelihood over all possible weights (marginalization) which gave the integral. The posterior distribution, $p(w|x,y)$ is thereafter obtained using Bayesian approximation procedures (variational inference). This can be likened to the use of dropout units for model regularization, which randomly sets weights to 0 or 1 in every weight layer during implementation. This method involves fitting a simple distribution $q*(w)$ parametrized by θ to the posterior distribution, replacing the intractable marginalization with an optimization task over the parameters of the simple distribution; thereby minimizing the KL divergence between $q*(w)$ and the true model posterior $p(w|x,y)$. Since we are interested in making our approximation distribution as close as possible to the posterior distribution, we defined approximating variational distribution q_w that can be evaluated analytically, thus minimizing the KL divergence between the variational distributions and posterior distribution: $KL(q_w||p(w \mid x,y)$, to obtain an approximating predictive distribution:

$$p(y^*/x,x,y) \approx \int p(y^*/x^*,w)q^*_\theta(w)dw =: q^*_\theta(y^*/x^*) \tag{4}$$

and its loss function would be,

$$L(\theta,p) = -\frac{1}{N}\sum_{i=1}^{N} log.p\Big(y_i|f^{w_i}(x_i)\Big) + \frac{1-p}{2N}\Big||\theta|\Big|^2 \tag{5}$$

3 Methodology

Modeling a dynamic network environment, such as smart grid cyber-attack intrusion, deserves a proactive approach considering its capabilities and operational characteristics. Smart grids are a system-of-systems that requires a comprehensive analysis of its efficient and secure infrastructures (devices, network protocols, and software application). A generic framework for performance optimization in terms of threat detection and classification is developed (see figure 3), with its detection procedure in figure 4. The threat analysis process started with a mathematical framework for measurement of uncertainties in smart grid network based on Bayesian probability distribution. This is useful as our work entails characterizing the relationship between attack steps (pattern) and detectors. The interleaving of the Bayesian distribution and the neural networks (NNs) approach used in data learning and prediction is necessary because infinitely wide NNs with distributions placed over their weights easily converges to Gaussian processes [37]. This method has helped to contain the problem of overconfident predictions in regions of sparse data and overfitting in data training [38] even though it has challenging inference and computational cost.

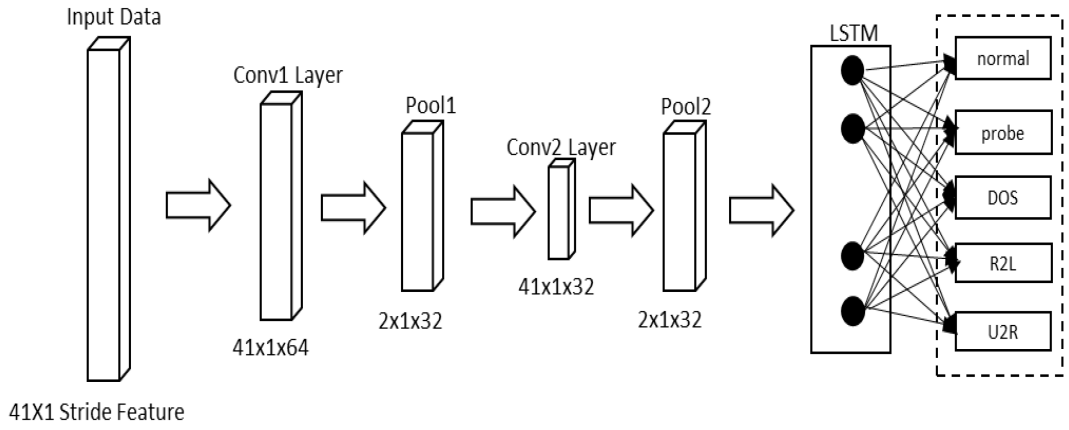

Figure 3: Architecture of the proposed neural network (SGtechNet)

3.1 Distribution of Weights in Layers

In this Bayesian neural network, everything, including the weights, are seen as probabilistic distributions. It is this that makes Bayesian networks unique. This distributions over weights can be made flexible by allowing only a distribution on certain level of subspace in each weight matrix to be learned instead of the entire weight matrices to reduce dimensionality [39]. The posterior distribution of different values of W can be computed as:

$$p(w = j/x_1, \ldots, x_p) = \frac{P(x_1, \ldots, x_p/w = j)p(w = j)}{P(x_1, \ldots, x_p)} \tag{6}$$

Applying Bayes' rule,

$$= \frac{(\Pi_{i=1}^{p} P(X_i/W = j))P(W = j)}{(PX_1, \ldots, X_p)} \tag{7}$$

If the conditional independence assumption is invoked,

$$p(w = j/X_1, \ldots, X_2) = \frac{1}{\alpha}(\Pi_{i=1}^{p} p(x_i/w = j))p(w = j), 1 < j < m \tag{8}$$

where α is a proportionality constant independent of j. This implies that $P(W = j/X_1, \ldots, X_p)$ can be computed as being proportional to a product of simpler individual terms $p(x_i/w = j)$ and, $p(w = j)$. To evaluate the posterior probability distribution in the multi-modal case, expressions for the prior distribution $p(w = j)$ and likelihood function $p\left(\frac{W=j}{X_{(1,\ldots,X_P}}\right)$ will be needed. If we assume prior probability is a zero-mean Gaussian function of weights of the form:

$$p(w = j) = \frac{1}{Z_{W=j(\alpha)}} Exp.(-\frac{\alpha}{2}||W = j||^2) \tag{9}$$

Normalizing factor $Z_{C=j(\alpha)}$ is given by

$$Z_{W=j(\alpha)} = \left(\frac{2\pi^{W=\frac{j}{2}}}{\alpha}\right) \tag{10}$$

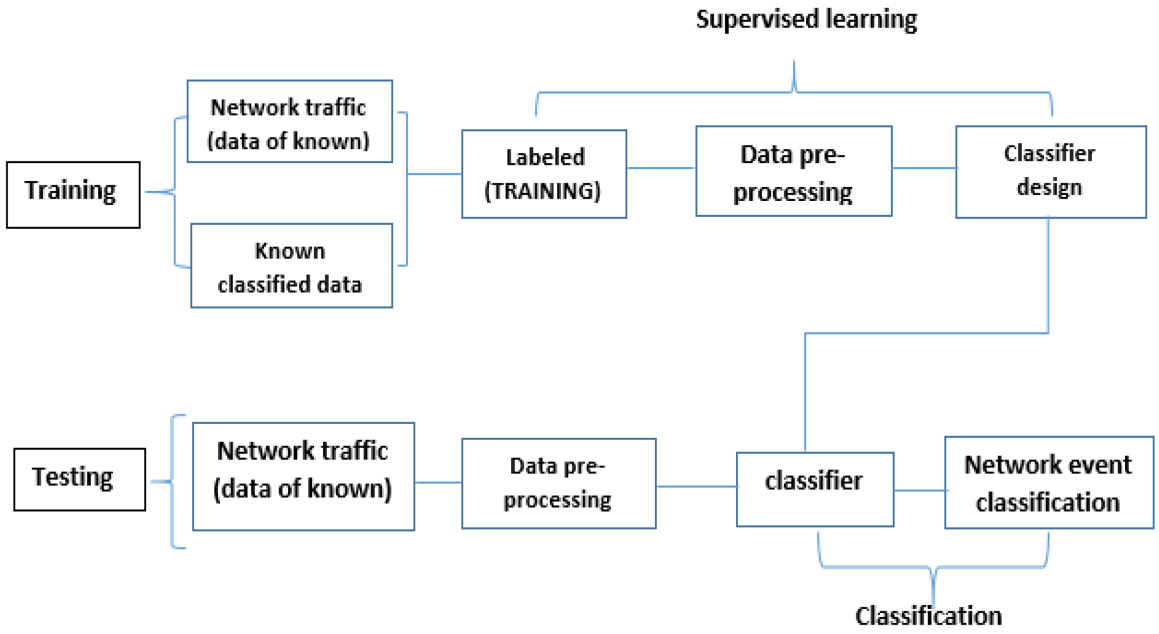

Figure 4: Proposed intrusion detection process

The performance of multinomial classification was evaluated by [40] using multilayer perceptron networks to assign classification probabilities where the first layer of the neural network contains the input comprising 41 features and the final layer containing the output (for both binary and multinomial class categories). All the intervening hidden layers were connected to the nodes in the adjacent layers. In this case, variational inference performs the optimization over distributions instead of point estimates. Uncertainties are assumed to exist even in the weights, so this network tends to model the weights as distributions and a prediction of output (y) is done through optimization of the weights (via gradient descent and backpropagation) using a loss function, as shown in Equations 9 and 10. In practical terms, dropouts are used in such a way that a prior distribution is placed over the space of weights, integrating the weights to obtain a marginal probability for each set of corresponding network outputs. This can be simply interpreted as performing approximate inference in a probabilistic interpretation of neural network model.

3.2 Data Description and Pre-Processing

This work leveraged on the network intrusion simulated dataset NSL-KDD; an improved version of KDD CUP'99 from the MIT lab repository as a benchmark dataset to train and test our network that contained a synthetic attack data. The data contained 21 predicted labels records with 41 features for attacks and normal. Each feature has 3 types of attribute values (normal, binary, and numeric). Attack classes were categorized into DoS, Probing, U2R, and R2L. NSL-KDD contains about 75% reduction in redundant recorded in the popular KDD'99 test set and have sufficient training set that includes attack-type labels and difficulty level. The training set has a total of 125,973 records comprising attacks and normal, and testing set has a total of 22,544 records for both attacks and normal as well. It contained a total of 24 training attack types, with an additional 14 additional attack types introduced in the testing data were not from the same probability distribution as the training data, making the task a more realistic one since novel attacks are believed to be variants of known attacks which means the signature of known attacks

is enough to detect novel variants [41]. However, the dynamics of cyber-attacks on smart grids creates attack patterns have complex characteristics. Therefore, the first step in this study was to extract attack features that model detection of novel attacks in grid network. To achieve this, features were analyzed in both time and frequency domains. The essence was to enhance the feature extraction process that would accentuate the signature of various attack types. Empirical observations during the exploratory data analysis showed that time affects some traffic-based features as well as frequency of occurrence on most of the content-based features.

3.3 Model Design

The computational method to predict the likelihood of a given data sample belonging to a predefined class i.e., normal or attack, was achieved using a Bayesian neural network with deep learning algorithms. This supervised learning approach to intrusion detection considered four different complementary threat perspectives in software-defined network-enabled smart grid: method, target, software, and identity. Features indicating network intrusions were carefully selected from the TCP dump and analyzed against novel invariants. The dataset used for both training and testing of the model contained both continuous and categorical data.

3.3.1 Network Architecture

Smart grid mesh networks connect remote smart meter mesh networks to the utility through either wired or wireless infrastructure. The level of complexity and volatility of this cyber-physical system is significant. In this context, we look to the deep network-like ensemble method only inasmuch that no combination of predictions from individual classifiers and weights are modeled as a distribution. To achieve major objectives of this study, which is reducing model's size and overfitting, approximation distributions were placed over the kernels before the convolutional process.The down-sampling operation, using pooling size 2, worked very well in reducing the model size (by 4 times) compared to existing architectures, without necessarily affecting the model accuracy. To capture the spatiotemporal series patterns over different time-steps of newly formed features emanating from Maxpooling operation in convolutional layer, the newly constructed feature map vector (see Figure 5) is fed to LSTM and GRU algorithms to learn the long-range temporal dependencies.

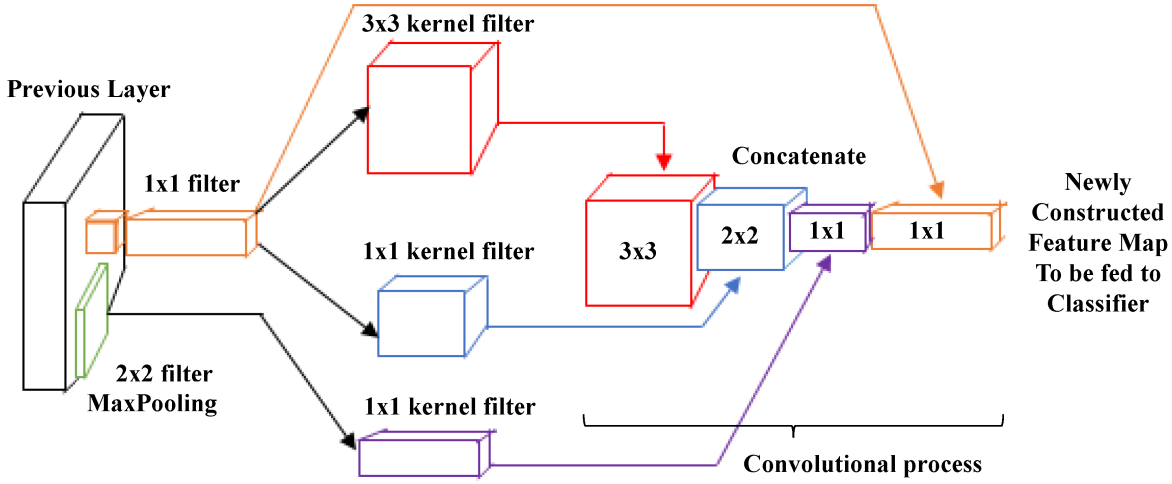

Figure 5: Spatiotemporal Feature Extraction Module

This data learning process was made in such a way that the output of the first CNN network data compression process became an input to the LSTM in a serial manner, as shown in Figure 3, although not without a data reshaping process. The network was modeled using CNN-LSTM and validated on GRU to ascertain the efficiency of both classifiers and their effectiveness as relates to the depth and complexity of symbolic sequences.

3.3.2 Model Training

For the purposes of probabilistic modeling, binary variables for each input point were sampled for each network unit in each layer (excluding the last layer). Each binary variable takes value 1 with probability p_i for layer i, dropping a unit (i.e., setting its value to zero) where binary variable takes value 0. In principle, this placed a prior distribution over each kernel which sets the kernels to zero for different patches, and approximately integrates each kernels-patch pair with Bernoulli variational distributions. Dropout were used after every convolution layer before the pooling operation and approximation of the predictive distribution helped immensely in its scaling. During network parameter tuning of the model for optimal performance, various network topologies with varying filters and lengths were experimented with before choosing the 64-kernel size with a min batch size of 128, as shown in Table 1. The optimal value of the regularization hyperparameter (λ) was found to be 0.5 due to the largeness of the network. Reducing the activations of the layers by half facilitated the learning process and allowed the network to learn different redundant representations. Validation testing, using the GRU algorithm, had a dropout optimal value of 0.1. The learning rate as shown in Table 2 was fixed at 0.001 throughout the training period over 20 epochs and EarlyStopper was set to terminate training after five (5) consecutive decreases in validation accuracy.

Two activation functions, ReLU and Softmax, were used at different points. ReLU was used in the hidden layers due to its computational efficiency and Softmax was used in the last layer of the network because of its doggedness in defining the probability distributions over classes. This made it very useful especially in the multinomial classification performed. We tested the adaptability of the classifiers in detecting novel variants by introducing some specific attacks into the holdout data that were not in the training data and then performing different categories of classifications.

CNN-LSTM	GRU
Conv1D (Output 64-Kernel3)	GRU (Output 64)
Conv1D (Output 64- Kernel3)	Dropout (0.1)
MaxPooling1D (Pool size=2)	GRU (Output 64)
Conv1D (Output 128- Kernel 3)	Dropout (0.1)
Conv1D (Output 128- Kernel 3)	GRU (Output 64)
MaxPooling1D (Pool size=2)	Dropout (0.1)
LSTM (64)	GRU (Output 64)
Dropout (0.5)	Dropout (0.1)
Dense (5)	Dense (5)

Optimizer	Min-batch size	Learning rate	Epoch size	Activation function
Adam	128	0.001	20	Rectified linear unit (ReLU)

3.4 Implementation

We created a training dataset as X and loaded its label as target Y and fed it through the network. The network estimates each current state variables with uncertainties and loss generated is used for the computation of the gradient. The network weights for the model were then adjusted continually until a better gradient was obtained. In principle, the approximations to the model posterior with variational inference was achieved by adding dropout layers after certain weight layers in our network. By this method, our Bayesian neural network reduces to performing a dropout after every layer with an approximating distribution in the cause of networking training.

3.4.1 Experiments

Figure 5 showed how the number of feature vectors from previous convolution was convolved after the MaxPooling layer and compressed by half using a 1x1 kernel size filter placed before a 1x1 and 3x3 kernel filters. It convolutes with a smaller number of features and their output filter banks concatenated into a single output vector before recursively expanding for the next layer. This convolution process created the new feature maps fed to the classifier. Implementation was done in an NIVIDIA GeForce GTX1080 TI GPU enabled TensorFlow. Intel (R) Xeon (R) E5-2683 v3 @ 2.00GHz 56 Core CPU, 64 GB RAM and NVIDIA Tesla K80 GPU running on the Ubuntu Server 16.04.3. Because our network needs to match the prior distribution for both training and testing data, the Adam optimizer was used for updating the posterior distributions using the prior distribution and the likelihood until they converge. From table 1, our proposed model and its comparator share similar parameters. The model is tasked with simultaneous classification of invariants into binary class (normal or attack) and multinomial class.

Attack	Precision		Recall		F1-score		Support	
	CNN-LSTM	GRU	CNN-LSTM	GRU	CNN-LSTM	GRU	CNN-LSTM	GRU
0	1.0000	0.9978	0.9999	0.9087	0.9999	0.9512	3925650	250436
1	0.9999	0.7244	0.9998	0.9917	0.9999	0.8373	972781	60593
Accuracy	CNN-LSTM				GRU			
	0.99988				0.99975			

Attack	Precision		Recall		F1-score		Support	
	CNN-LSTM	GRU	CNN-LSTM	GRU	CNN-LSTM	GRU	CNN-LSTM	GRU
0	0.9985	0.9988	0.9077	0.9067	0.9509	0.9505	250436	250436
1	0.7226	0.7208	0.9943	0.9955	0.8370	0.9283	60593	60593
Total Accuracy	CNN-LSTM				GRU			
	0.9267				***0.9286***			

Attack	Precision		Recall		F1-score		Support	
	CNN-LSTM	GRU	CNN-LSTM	GRU	CNN-LSTM	GRU	CNN-LSTM	GRU
0	0.9999	0.9999	0.9997	0.9997	0.9998	0.9998	3925650	3925650
1	0.9987	0.9987	0.9996	0.9996	0.9997	0.9991	972781	972781
Accuracy	CNN-LSTM				GRU			
	0.9997				0.9996			

Attack	Precision		Recall		F1-score		Support	
	CNN-LSTM	GRU	CNN-LSTM	GRU	CNN-LSTM	GRU	CNN-LSTM	GRU
0	0.9988	0.9999	0.9067	0.9997	0.9505	0.9998	250436	3925650
1	0.7208	0.9987	0.9955	0.9996	0.8362	0.9991	60593	972781
Accuracy	CNN-LSTM				GRU			
	0.9249				*0.9282*			

4 Results Analysis

Performance analysis of this study is based on accuracy as a quintessential evaluation metrics suitable for both binary and multiclass classification especially on models having same sample and model configuration. Based on the validation accuracy results shown in Figures 11 and 13, CNN-LSTM and GRU classifiers performed well.Almost all the models experienced an unstable training trajectory, and it was observed that the validation of the best performing model i.e., GRU, attained the Point of Diminishing Returns between epochs 16 and 20 at the 2-class and 5-class categories, respectively. This fluctuation occurring after the Point of Diminishing Returns could be likened to overfitting resulting from dropouts applied after convolutions. However, this was contained by using predictive distribution from samples, in the region of space surrounding the synthetic data that was later tested. However, the highest training loss was recorded in the 5-class category. CNN-LSTM had the best training accuracy in both categories tested. Figure 6 shows the training and validation accuracies of the CNN-LSTM classifier in the binary classification with the highest accuracy recorded immediately after commencement of the model validation, towards epoch 1. As the training accuracy increased over the increasing epochs, the validation accuracy decreased over five consecutive epochs, resulting in abrupt termination. Similarly, Figure 7 depicts its training and validation accuracies in multiclass. Figure 8 shows training and validation accuracy of the GRU classifier in the binary class category. The training accuracy increased successively together with the validation accuracy until epoch 6 when it became unstable because of suspected overfitting. Even though GRU accuracy fluctuated in the 2-class category, it became more stable in the 5-class.

Furthermore, the validation accuracy of the GRU classifier outperformed CNN-LSTM in both classification categories (see Figure 10 and 11). In Figure 9, the validation accuracy of the GRU multiclass increased correspondingly with the training accuracy until epoch 19, where it decreased slightly before the termination point. Figure 10 and 11 respectively showed the validation accuracies in the binary and multiclass classification category for the models. $R2$ value of figure 6 for CNN-LSTM showed that more than half of observed variation can be explained by the model's inputs from holdout dataset used for model validation. GRU has a relatively low and stable $R2$ in both test categories. Adaptiveness of this network architecture to other areas of potential attacks other than smart grid network was tested by introducing a synthetic attack features depicting attacks from real-time internet environment. The accuracy achieved was almost the same with that of NSL-KDD dataset.

4.1 Discussion

One major limitation of the model compression technique implemented is that, as the model size is reduced, the number of parameters slightly increased which results in a marginal increase in financial burden in terms of implementation. Therefore, further work is proposed to develop a systemic method of reducing the model size without necessarily increasing the number of model parameters. We would expect that this will guarantee the cost effectiveness and light weight expected of a good model. The compression was achieved by modifying the hidden layer of the network through the addition of a 1x1 kernel size filter before the 3x3 kernel size filter that was used in the baseline model, and the output of

the layers was squeezed by half (1/2). Instead of using 1x1 and 3x3 convolutions like [42] for feature extraction 1x1 and 1x3 kernel filters were used before convolution operation and another 1x1 and 1x1 used before, and after convolutions respectively as shown in figure 5. However, this approach has not changed the size of the data, but rather has simply compressed it to fit. By this convolutional process, sequential data is thoroughly learned, and feature map created. The choice of small filters is to reduce both the number of network parameters and the processing time. Large filters are most appropriate in image data where it is necessary to determine the central position of the image. Furthermore, due to the fact the frequent problem of misclassifications in image and time series problems are the result of inappropriate feature encoding and vectorization, the data mining aspect of this work was thoroughly handled.

4.2 Chi-Square test

We used the Chi-Square statistical analysis tool to determine the degree at which the training accuracy differed from the validation accuracy and to further investigate how additional features in the test data affected the classification accuracy in the multinomial class category. This classical tool is mostly used in complex data analysis due to its precision. The Chi-Square test was conducted using a spreadsheet developed by the Mathematics, Science and Technology College of the Education University of Illinois, USA[].Experimental results were approximately distributed as Chi-Square samples.

$$Chi - Test = \sum \frac{(E-0)2}{E} \tag{11}$$

where E is expected, while O is obtained. The Chi-Square results shown in Table 7 indicate that the discrepancy between the training and validation accuracies is not statistically significant. It also shows that the frequency at which the training and validation outcomes could differ sufficiently to produce values that are as large or larger than the Chi-Square value per model, is highest in both GRU test categories: 3.7 and 9.2 respectively, with the lowest in the CNN-LSTM in 2-class (marked with asterisks). However, the overall percentage of difference is 10.7% for CNN-LSTM and 8.9% for GRU.

Models	2-class category		5-class category		Percent (%) difference
	Chi-square	Freq.	Chi-square	Freq.	
CNN-LSTM	0.032779	***0****	0.036496	***0.921448****	10.7311
GRU	0.033389	3.704944	0.036508	***9.221656****	8.92456

4.3 Comparative Analysis of Model

Table 8 compares our proposed model implemented in TensorFlow, using deep learning classifiers on the NSL-KDD, with the comparator model [41] (A Detailed Analysis of the KDD CUP 99 Data Set) based on K-means classifier, with spark on the NSL-KDD dataset. This is necessary because the model utilized linear combinations of models in the same way as in our method and was also tested on the same dataset. There were substantial reductions in terms of model size and processing time as well as marginal improvements in terms of validation accuracy. Importantly, [33] demonstrated how this response time could enable attackers to determine whether the target network is using SDN or not to leverage launching an attack. So, processing time is a good consideration for smart grid network intrusion detection model, and that justifies the computational speed of our proposed model.

Model Statistics	**Models**		
	Proposed Model		**KDD-Spark with K-means Classifier [26]** (Applying Convolutional Neural Network for Network Intrusion Detection)
Model Size	3.404MB		10.9G
Training Time	CNN-LSTM	43358s	6640.02s
	GRU	1447s	
Predicted Time	CNN-LSTM	80.59s	1090.93s
	GRU	15.55s	
Validation Accuracy (%)	CNN-LSTM	***92.67***	***92.64***
	GRU	***92.86***	

Performance Accuracies	**Datasets**			
	NSL-KDD		**Kyoto Honeypot [18]** (Network Intrusion Detection Systems in High-Speed Traffic in Computer Networks)	
Training	99.98%		93.29%	
Validation	***92.82%***		***81.54%***	
Prediction time	CNN-LSTM	***80.59s***	GRU+SVM	***82s***
	GRU	***15.55s***		

We went further to a performance comparison of our model with another work using a different dataset (Kyoto) as shown in table 9. The purpose was to demonstrate performance improvement using our model since both datasets share very similar attributes/features. The result was that our model showed proven superiority in both training and validation accuracies as well as computational time. And can easily detect sophisticated attacks such as "BlackEnergy" [**?**] etc. Comparing the performances of the two classifiers (CNN-LSTM and GRU) as shown in Table 8 and Table 9 is basically to determine whether increase in the model's depth results to better memorization. The analysis result showed that GRU and CONV-LSTM networks performed very well on low complexity and high complexity sequences respectively, as depicted in binary class and multiclass classification tasks.

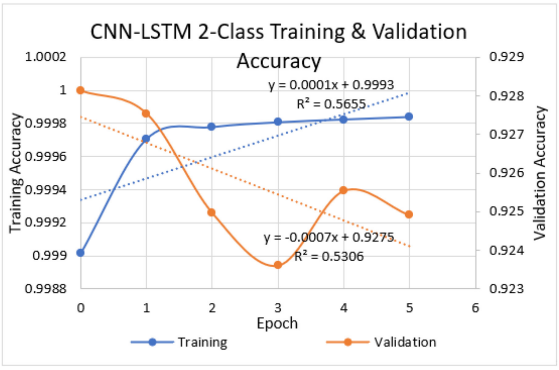

Figure 6: Plot of validation accuracy over training iterations in CNN-LSTM 2-class.

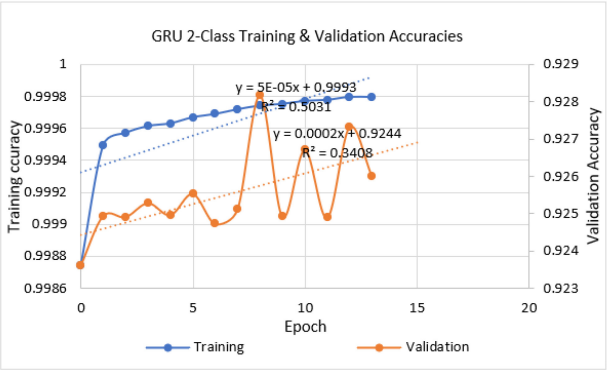

Figure 7: Plot of validation accuracy over training iterations in GRU 2-class

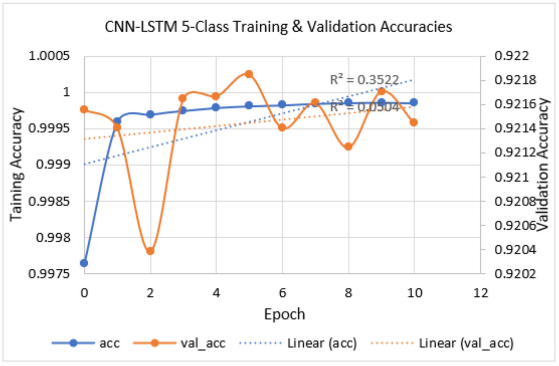

Figure 8: Plot of validation accuracy over training iterations in CNN-LSTM 5-class.

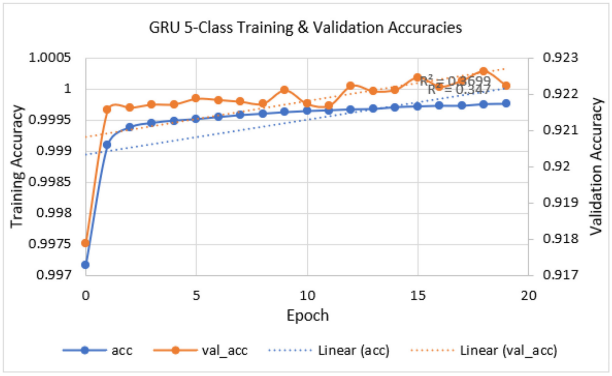

Figure 9: Plot of validation accuracy over training iterations in GRU 5-class

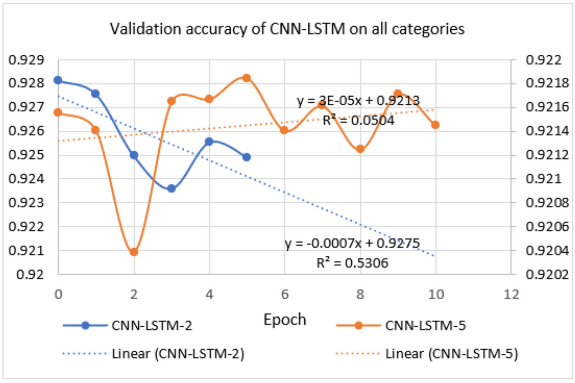

Figure 10: Plot of validation accuracy of CNN-LSTM in all the classes.

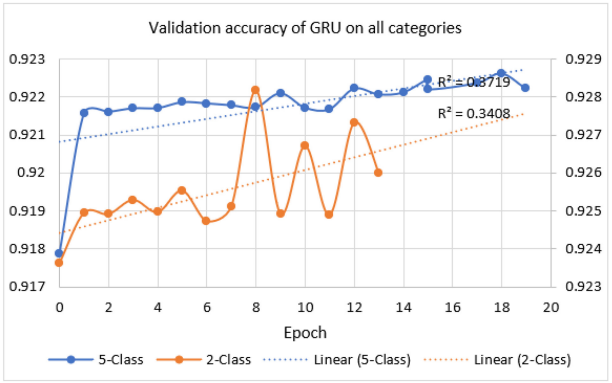

Figure 11: Plot of validation accuracy of GRU in all the classes

5 Conclusion

We investigated the effectiveness and adaptability of applying the Bayesian deep learning method in detecting and classifying attacks, termed as zero-day (unknown), capable of causing a systemic manipulation and forgery of meter measurements, or injection of falsified data that could compromise confidentiality of a smart grid network. Firstly, a smart grid network intrusions is modeled as a set of probability distribution uncertainties, and neural network-based architecture termed 'SGtechNet' developed to detect and classify potential attacks in two different categories. This model combined the flexibility, scalability, and probabilistic approach to model distribution over the neural networks models' weights by approximating the models' posterior probability with the variational inference. It was demonstrated empirically that this architecture can easily detect invariants and other resistant adversarial by approximating the prediction distribution over weights in such a way that posterior variational inference distribution is made to have little or no variation from the true model posterior. SGtechNet is tested on NSL-KDD and synthetic attack data, and the performance showed robustness and very adaptive to other threat environments that operates in a real-time. The model size reduction achieved by SGtechNet reduced the inference time and computer resources than the baseline model, without necessarily affecting the accuracy. Its robustness was demonstrated through its ability to classify threats in both the binary class and multi-classes with high-performance accuracy and high computational speed. The network was tested on CNN-LSTM and GRU classifiers which showed that the CNN-LSTM performed optimally in detecting unknown attacks in the binary class although it was surpassed by the GRU in terms of multi-class classification. The

overall comparative analysis showed that the GRU demonstrated the best validation accuracy in both attack categories. Further analysis to determine the level of variance between the training and validation accuracies, using Chi-Square test, showed that the variation between the training and validation results is statistically insignificant. The most significant outcome of this research, however, is the cost effectiveness of the model to real-time applications as well as its adaptiveness. SGtechNet is lightweight, so it is considered for complex network like smart grid requiring fast-to-evaluate approximation, to hasten the processing speed of the packets entering the network traffic. In the future, we hope to detect invariants using SGtechNet in a smart grid network traffic modeled outside adhoc networks to mitigate the challenges encountered in periodic traffic flow.

References

[1] A. Hansen, J. Staggs, and S. Shenoi. Security analysis of an advanced metering infrastructure. *International Journal of Critical Infrastructure Protection*, 18:3–19, September 2017.

[2] Symantec Corporation. *Internet Security Threat Report 2017*. Symantec Corporation USA, 2017.

[3] H. Orman. The morris worm: a fifteen-year perspective. *IEEE Security & Privacy*, 1(5):35–43, September-October 2003.

[4] H. T. Mouftah and M. Erol-Kantarci. *Smart Grid: Networking, Data Management, and Business Models*. CRC Press, 2017.

[5] Q. Wang, W. Guo, and K. Zhang. Adversary resistant deep neural networks with an application to malware detection. In *Proc. of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD'17), Washington , District of Columbia, USA*, pages 1145–1153. ACM, August 2017.

[6] W. Ma, Y. Wu, F. Cen, and G. Wang. Mdfn: Multi-scale deep feature learning network for object detection. *Pattern Recognition*, 100:107149, December 2019.

[7] D. Li, L. Deng, M. Lee, and H. Wang. Iot data feature extraction and intrusion detection system for smart cities based on deep migration learning. *International Journal of Information Management*, 49:533–545, December 2019.

[8] G. M. U. Din and A. K. Marnerides. Short term power load forecasting using deep neural networks, January 2017.

[9] Cylance Data Science Team. *Introduction to Artificial Intelligence for Security Professionals*. Cylance Press, 2017.

[10] M. K. Putchala. Deep learning approach for intrusion detection system (ids) in the internet of things (iot) network using gated recurrent neural networks (gru). Master's thesis, Department of Computer Science & Engineering Wright State University, July 2017.

[11] A. A. Diro and N. Chilamkurti. Distributed attack detection scheme using deep learning approach for internet of things. *Future Generation Computer Systems*, 82:761–768, May 2018.

[12] P. Arun Raj Kumar and S. Selvakumar. Distributed denial of service attack detection using an ensemble of neural classifier. *Computer Communications*, 34(11):1328–1341, July 2011.

[13] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):1–58, July 2009.

[14] J. Lansky, S. Ali, M. Mohammadi, M. K. Majeed, S. H. Taher Karim, S. Rashidi, M. Hosseinzadeh, and A. M. Rahmani. Deep learning-based intrusion detection systems: A systematic review. *IEEE Access*, 9:101574–101599, January 2021.

[15] K. C. Laudon and J. P. Laudon. *Information System and the Internet: A problem-solving approach*. Dryden Press, 1998.

[16] M. Nakip and E. Gelenbe. Mirai botnet attack detection with auto-associative dense random neural network. In *Proc. of the 2021 IEEE Global Communications Conference (GLOBECOM'21), Madrid, Spain*, pages 01–06. IEEE, December 2021.

[17] P. Prasse, L. Machlica, T. Pevn´y, J. Havelka, and T. Scheffer. Malware detection by analysing network traffic with neural networks. In *Proc. of the 2017 IEEE Security and Privacy Workshops (SPW'17), San Jose, California, USA*, pages 205–210. IEEE, May 2017.

[18] J. Grimmer. An introduction to bayesian inference via variational approximations. *Political Analysis*, 19(1):32—-47, February 2011.

[19] P. Smyth. Note set 2: Multivariate probability models. Technical Report 14, Department of Computer University of California, 2022.

[20] P. Hennig and C. J. Schuler. Entropy search for information-efficient global optimization. *The Journal of Machine Learning Research*, 13:1809—-1837, March 2012.

[21] J. M. Henrández-Lobato, M. W. Hoffman, and Z. Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In *Proc. of the 27th International Conference on Neural Information Processing Systems (NIPS'14), Cambridge, Massachusetts, USA*, volume 1, pages 918—-926. MIT Press, December 2014.

[22] L. Vokorokos and A. Bal´á´ž. Host-based intrusion detection system. In *Proc. of the 14th IEEE International Conference on Intelligent Engineering Systems (INES'10), Las Palmas, Spain*, pages 43–47. IEEE, May 2010.

[23] M. Kumar, M. Hanumanthappa, and T. V. Suresh Kumar. Intrusion detection system for grid computing using snort. In *Proc. of the 2012 International Conference on Computing, Communication and Applications (ICCCA'12), Dindigul, India*, pages 1–6. IEEE, February 2012.

[24] W. Bulajoul, A. James, and M. Pannu. Network intrusion detection systems in high-speed traffic in computer networks. In *Proc. of the 2013 IEEE 10th International Conference on e-Business Engineering (ICEBE'13), Coventry, UK*, pages 168–175. IEEE, September 2013.

[25] Y. Meng and W. Li. Adaptive character frequency-based exclusive signature matching scheme in distributed intrusion detection environment. In *Proc. of the 11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom'12), Liverpool, UK*, pages 223–230. IEEE, June 2012.

[26] S. Khan, K. Kifayat, A. K. Bashir, A. Gurtov, and M. Hassan. Intelligent intrusion detection system in smart grid using computational intelligence and machine learning. *Transactions on Emerging Telecommunications Technologies*, 32(6):e4062, June 2021.

[27] M. Yousefi-Azar, V. Varadharajan, L. Hamey, and U. Tupakula. Autoencoder-based feature learning for cyber security applications. In *Proc. of the 2017 International Joint Conference on Neural Networks (IJCNN'17), Anchorage, Alaska, USA*, pages 3854–3861. IEEE, May 2017.

[28] H. Z. Zhao, F. X. Liu, and L. Y. Li. Improving deep convolutional neural networks with mixed maxout units. *PLoS One*, 12(7):1–16, July 2017.

[29] K. S. Anil Kumar; V. Nanda Mohan. Adaptive fuzzy neural network model for intrusion detection. In *Proc. of the 2014 International Conference on Contemporary Computing and Informatics (IC3I'14), Mysore, India*, pages 987–991. IEEE, November 2014.

[30] T. Le, J. Kim, and H. Kim. An effective intrusion detection classifier using long short-term memory with gradient descent optimization. In *Proc. of the 2017 International Conference on Platform Technology and Service (PlatCon'17), Busan, Korea (South)*, pages 1–6. IEEE, February 2017.

[31] J. Kim, J. Kim, H. L. T. Thu, and H. Kim. Long short term memory recurrent neural network classifier for intrusion detection. In *Proc. of the 2016 International Conference on Platform Technology and Service (PlatCon'16), Jeju, Korea (South)*, pages 1–5. IEEE, February 2016.

[32] A. F. M. Agarap. A neural network architecture combining gated recurrent unit (gru) and support vector machine (svm) for intrusion detection in network traffic data. Technical Report 5, Cornell University, 2017.

[33] V. Ravi, S. K. Padannayil, and P. Poornachandran. Applying convolutional neural network for network intrusion detection. In *Proc. of the 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI'17), Udupi, India*, pages 1222–1228. IEEE, September 2017.

[34] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Proc. of the 31st International Conference on Neural Information Processing Systems (NIPS'17), Red Hook, NewYork, USA*, pages 6405—-6416. Curran Associates Inc., 2017.

[35] Y. Gal and Z. Ghahramani. Bayesian convolutional neural networks with bernoulli approximate variational inference. In *Proc. of the International Conference on Learning Representations (ICLR'16), Caribe Hilton, San Juan, Puerto Rico*, pages 1–12. arXiv, May 2016.

[36] D. J. C. MacKay. Bayesian interpolation. *Neural Computation*, 4(3):415—-447, May 1992.

[37] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proc. of the 33rd International Conference on International Conference on Machine Learning (ICML'16), New York, USA*, volume 48, pages 1050—-1059. PMLR, June 2016.

[38] B. M. Christopher. Bayesian methods for neural networks. Technical Report NCRG/95/009, Technical Report NCRG/95/009, Neural Computing Research Group, Aston University, 1995.

[39] M. W. Dusenberry, G. Jerfel, Y. Wen, Y.-A. Ma, J. Snoek, K. Heller, B. Lakshminarayanan, and D. Tran. Efficient and scalable bayesian neural nets with rank-1 factors. In *Proc. of the 37th International Conference on Machine Learning (ICML'20), online*, pages 2782–2792. PMLR, August 2020.

[40] T. Auld, A. W. Moore, and S. F. Gull. Bayesian neural networks for internet traffic classification. *IEEE Transactions on Neural Networks*, 18(1):223–239, January 2007.

[41] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani. A detailed analysis of the kdd cup 99 data set. In *Porc. of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA'09), Ottawa, Ontario, Canada*, pages 1–6. IEEE, July 2009.

[42] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and ¡0.5mb model size. In *International Conference on Learning Representations (ICLR'17), Palais des Congrès Neptune, Toulon, France*. arXiv, February 2017. `https://doi.org/10.48550/arXiv.1602.07360`.

Author Biography

Alexander N. Ndife received a bachelor's degree in Electrical/Electronic Engineering from Anambra State University, Nigeria in 2008 with specialty in telecommunications, Masters' degree in Electronics and Computer Engineering (Communications) from Nnamdi Azikiwe University, Nigeria in 2014, and subsequently Ph.D. degree in Smart Grid Technology from Naresuan University, Thailand. He is a Data Scientist, and his research interests include Wireless Networks, Artificial Intelligent Systems, Deep Learning, Image Processing, and Cybersecurity & Smart Grid Networks. He is a registered engineer in Nigeria and CEO Ogugbalex Engineering limited. He has attended various academic conferences and seminars, published numerous scientific papers and a member of various engineering organizations including Nigerian Society of Engineer (NSE), IEEE, and International Association of Engineers (IAENG).

Yodthong Mensin obtained bachelor's degree (B. Sc) in Computer Science, and Master (M. Sc) in Information Technology from Naresuan University, Thailand. He obtained Doctor of Philosophy degree in Energy, communities, and the environment from Chiang Mai Rajabhat University.He also received the Cert. in Renewable Energy Technology from the University of Applied Science, Stralsund, Germany. He is presently a deputy Director of research and academics affairs in School of Renewable Energy and Smart Grid Technology (SGtech), Naresuan University, Thailand.He is a permanent speaker for IEEE conference in the section of Power and Energy Society (PES) in Thailand. He has experience in the field of a microgrid system, automated demand response (ADR), virtual power plant (VPP), energy management system, smart grid data utilization, and energy trading platform with

blockchain technology for more than 15 years. Currently, he is the advisory's team member for the mega-project implementation of Thailand utilities, ministry of energy and private sector.

Wattanapong Rakwichian graduated from Srinakharinwirot University, Phitsanulok, Thailand in Physics (B. Ed.). He obtained master's degree in physics from Chiang Mai University, Thailand, and Doctor of Philosophy (Ph. D) in Bioregulation - Renewable Energy from Tokyo University of Agriculture, Tokyo, Japan. He is presently the Director of School of Renewable Energy and Smart Grid Technology (SGtech), Naresuan University, Thailand. He had served in various positions in recent times including Sub-committee of the Royal Thai Project on Renewable Energy, Consultant of Ministry of Energy, Ministry of Science Technology and Energy, etc. He has written several textbooks on Physics, Mathematics, Solar Energy, Digital Systems, etc. He has also written many research articles for international journals and proceedings, hosted, and participated in many national and international conference related to science, solar energy, and other energy sources.

Paisarn Muneesawang received a B.Eng. Degree in Electrical Engineering from the Mahanakorn University of Technology, Bangkok, Thailand, in 1996, M.Eng. Sci. Degree in Electrical Engineering from the University of New South Wales, Sydney, NSW, Australia, in 1999, and Ph. D from the School of Electrical and Information Engineering, University of Sydney, Sydney. He was a Post-Doctoral Research Fellow with Ryerson University, Toronto, ON, Canada, from 2003 to 2004, and an Assistant Professor with the College of Information Technology, University of the United Arab Emirates, Al Ain, UAE, from 2005 to 2006. He has been a Visiting Professor with Nanyang Technological University, Singapore, since 2012, and with Ryerson University, Toronto, Canada, since 2013. He was the Vice President for Administrative Affairs, Naresuan University, Phitsanulok, Thailand, where he is currently a Professor and Dean of the Graduate School. He co-authored Multimedia Database Retrieval: A Human-Centered Approach (Springer, 2006) and Unsupervised Learning - A Dynamic Approach (Wiley-IEEE Press, 2013). He co-edited Advances in Multimedia Information Processing - PCM 2009 (Springer, 2009). His current research interests include multimedia signal processing, computer vision, and machine learning. He has served as the Registration Co-Chair of the International Conference on Multimedia and Expo 2006 and the Technical Program Co-Chair of the Pacific-Rim Conference on Multimedia 2009.