Long Short Term Memory and Gated Recurrent Unit Predictive Models for Industrial Control Systems

Shraddha Suratkar, **Mrunmayee Dhapre and Apurva Vaje**

Department of Computer Engineering
Veermata Jijabai Technological Institute
Mumbai, India
sssuratkar@ce.vjti.ac.in, mvdhapre_b16@ce.vjti.ac.in and amvaje_b16@ce.vjti.ac.in

ABSTRACT

As Operational Technologies (OT) and Internet of Things (IoT) grew popular, Industrial Control Systems (ICS) which are frequently managed through a Supervisory Control and Data Acquisition (SCADA) systems gained importance but simultaneously anomalies in ICS became a security concern. This paper presents a data-driven approach of predictive modeling for Energy Management System logs that exploits the relationship between data elements in the logs and the predictable aspect of communication patterns between devices in ICS networks using the time series structure of their logs. Specifically, two Recurrent Neural Networks - Stacked Long Short Term Memory (LSTM) and Stacked Gated Recurrent Unit (GRU) models - have been employed to model the behavior of these logs and comparison between these models is demonstrated. Various measures like accuracy, loss, memory usage and testing time are implemented to compare the performance of the models.

Keywords: Industrial Control System; Energy Management System; Data-driven Modeling; Deep Learning; Recurrent Neural Network; Long Short Term Memory; Gated Recurrent Unit.

2020 Mathematics Subject Classification: 68T07, 68T20, 68U99

2012 Computing Classification System:

- **Security and privacy∼Intrusion/anomaly detection and malware mitigation∼Intrusion detection systems**
- **Computing methodologies∼Modeling and simulation∼Model development and analysis**
- **Computing methodologies∼Machine learning∼Machine learning approaches∼Neural networks**

1 Introduction

Digital transformation connects Operational Technology like never before. That connectivity makes industrial organizations smarter and safer, gives them better management, greater visibility and improves uptime and productivity. But modern plans have a high degree of interconnectivity which opens the Operational Technology (OT) environment to outside threats. Reference(Allhoff, F., and Henschke, A., 2018) discusses ethical issues related to IoT. An OT cyberattack affects more than just data. It can cause physical harm to personnel, environment and industrial equipment, and thus, disrupt productivity. OT professionals are at a disadvantage because they are at the beginning of the security journey. To safeguard their environment, they need a total security program that helps them understand the risks, improve asset visibility across the organization and remedy the impact of cyberattacks(Murray, G. et al., 2017).

Energy production, large scale manufacturing and all forms of automated processing have become critically dependent on digital communications and computer networking abilities. Information Technology is appearing throughout the operational space by the way of devices like smart meters, automated asset distribution systems and self-monitoring transformers, because of increasing wired and wireless communication between the growing number of smart devices. Modernization of OT through IT integration brings with it the required consideration of security. OT/IT convergence empowers more target control and monitoring with an easier investigation of data from network systems anywhere in the world.

Industrial Control Systems (ICS) which are an indispensable element of OT, are made up of the machines, systems, networks, and controls used to manage and automate industrial activities, improving their efficiency and lowering their cost (Kargl, F. et al., 2014). ICS functions vary according to the industry where they are applied. It is an umbrella term consisting of Distributed Control System (DCS), confined to a single location, like a certain factory, as well as, SCADA, used for systems that are scattered over a considerable topographical region, like a power grid. Its components - Programmable Logic Unit (PLC), fieldbus, human machine interface (HMI), workstation, etc - are explained in brief in (Chapman, J. P. et al., 2016). It also provides an analysis of several security measures for ICS. Reference(Stouffer, K. et al., 2015) also summarizes the security mechanisms present in such systems. A survey of analysis and design of fuzzy control systems for industrial applications is presented in (Precup, R., and Hellendoorn, H., 2011) and a different method for stable design of fuzzy logic control systems for chaotic processes is introduced in (Precup, R., and Tomescu, M., 2015).

Supervisory Control And Data Acquisition system (SCADA) gathers information from devices and sensors positioned at a remote location in the field and transmits it to the main station. The collected data are observed and analyzed on the SCADA connected computers in the main station. Based on this analysis, commands given by the computer or the operator can be relayed to control devices present at a substation in the field, also known as field devices. SCADA systems generally function with little human intervention and provide real-time environment supervision. This is possible because of the periodic acquisition of data like meter readings, sensor status, etc. Reference(Sayed, K., and Gabbar, H.A., 2017) elaborates this well. A real-time knowledge generation component is designed in (Skripcak, T., and Tanuska, P., 2013) to store and process these continuous values. An intelligent air quality analysis system implemented by integrating an artificial neural networks, for air pollutants concentrations forecasting, and rule-based expert systems for the analysis of air quality is presented in (Oprea, M., 2012).

One such avenue where SCADA can be adopted for system optimization is *Energy Management Systems (EMS)*. With the development of technology, electrical energy has emerged as an essential for the socio-economic growth of society. SCADA monitoring helps in increasing the energy efficiency of EMS. Such a SCADA-enabled EMS is described in (Mesaric, P., and Palasek, B., 2014), which demonstrates the architecture for incorporation of the sophisticated networking technologies with the conventional SCADA software.

2 Literature Survey

Various papers about Energy Management System applications of SCADA and other related topics have been studied. Renewable Energy Management System along with its structure and implementation using SCADA technology is illustrated in (Dumitru, C. D., and Gligor, A., 2012). Much research is going on in enhancing EMS that employs SCADA and making them smarter. Reference(Teixeira, A. et al., 2011) explores the security of SCADA-enabled EMS from intelligent attacks. A novel framework for an intelligent dynamic energy management system that combines dynamic programming and reinforcement learning is introduced in (Venayagamoorthy, G. K. et al., 2016). Reference (Ashok, A. et al., 2014) suggests a game-theory approach to deal with coordinated cyberattacks in smart-grids.

Data-Driven Modeling (DDM) relies on the study of the data gathered from the system. A model is designed based on relationship between the state variables of the system(input and output) by making some assumptions about its physical behaviour. This approach has potential to advance much more than the conventional empirical one. It facilitates numerical predictions, reconstruction of highly nonlinear functions, data grouping, classification, etc. Many such applications in various scientific disciplines are elaborated in (Montáns, F. et al., 2019). They are also useful in finding solutions to *inverse problems*. This entails determining the unknown causes based on observation of their effects. Local search algorithms like tabu search and simulated annealing were implemented for the solutions to these implicit inverse problems in (Nino-Ruiz, E. et al., 2018). A similar approach is shown (Nino-Ruiz, E., and Yang, X., 2019) for non-linear data assimilation.

Deep Neural Networks (DNN) spearheads the data-driven methods and can be utilized for refining the EMS. The multiple layers in deep learning provide multi-level abstraction and help in identifying relationships between input and output as well as in discovering intricate patterns in large data sets. An analysis of this is found in (LeCun, Y. et al., 2015). One application of DNN in EMS is for Energy Load Forecasting. Reference(Amarasinghe, K. et al., 2017) investigates the effectiveness of using various DNN models - Convolutional Neural Network (CNN), LSTM, Support Vector Machine (SVM), etc - for forecasting energy load. Another paper that explores the application of deep neural networks in renewable energy systems for energy forecasting is (Wang, H. and Lei, Z. and Zhang, X. and Zhou, B., and Peng, J., 2019). A DNN architecture employing stacked auto-encoder (SAE) and stacked denoising auto-encoder (SDAE) has been projected in (Khodayar, M. et al., 2017) for ultra-short-term as well as short-term forecasting of wind speed, which shows that DNN outperforms ANN with shallow architecture and models complicated non-linear relationships. In (Kim, Y., and Kim, Y.-S, 2017), an intelligent video surveillance system approach is proposed by optimized neural network in machine learning classification model to develop an enhanced loitering detection scheme .

Recurrent Neural Networks (RNN) is a class of Deep Neural Networks that accepts time-series as input and hence has vast applications in sequential data. An implementation of RNN and its subclasses on event logs is explained in (Hinkka, M. et al., 2019). The paper illustrates predictive modeling as the event logs are input to predict the next event as output. Methods for attribute clustering and selection to create features are also explained. An RNN model, trained on previously recorded data, is applied to obtain the impedances of a dc power electronic system in (Xiao, P. et al., 2010). These impedance characteristics can be utilized in stability analysis of the power-electronics-based distributed power systems. An LSTM model for intrusion detection is illustrated in (Gao, J. et al., 2019) for SCADA systems. Reference (Shi, Z. et al., 2018) introduces an RNN model with a new evaluation index and the dragonfly algorithm for tuning. This model is used for interval forecasting in wind power systems, to compute the uncertainties in energy production.

Anomaly detection is the identification of data points, observations or events that do not conform to the regular pattern present in the logs. It is also helpful in behavior analysis. A hybrid approach of detecting anomalies by applying multi-start metaheuristic techniques and genetic algorithms is elaborated in (Ghanem, T. F. et al., 2015). In that paper,a number of detector having high anomaly detection accuracy were generated for large datasets and then detector reduction stage was invoked. Anomaly and attack detection models for IoT sensors have been implemented in (Hasan, M. et al., 2019) applying various Machine Learning approaches. Another paper (Hollingsworth, K. et al., 2018) inspects various DNN models for anomaly detection. An approach for automated error detection and isolation in computer-based manufacturing systems based on Deep Auto-Encoders (DAEs) is elaborated in (Iqbal, R. et al., 2019).

Modern RNN-based learning technologies are implemented to generate knowledge about abnormality that happened in the system by predicting the next event. This neural network-based approach is considered as one of the core parts of modern industrial information and control system. Reference(Nguyen, V. Q. et al., 2018) shows such a neural network used for anomaly detection or prediction. LSTM model is applied for this network. The anomaly detection system implemented in (Feng, C. et al., 2017) applies an interesting approach of using a multi-level system in which the first level identifies package content signatures and compares them to those stored in a database using Bloom filter. The second level applies stacked LSTM for predictive modeling on data to predict the next package signature.

The major contributions of this paper are:

- An effort to make ICS secure by generating data-driven predictive models for anomaly detection.
- Deep learning techniques, which are regularly favored for security in Information Technology as well as for Data-driven Modeling, are practiced in Operational Technology.
- An approach to anomaly detection that takes advantage of the predictable pattern in event logs.
- A comparison between LSTM and GRU models over an ICS dataset.

3 Methodology

3.1 Problem Statement

Anomaly detection is employed for identifying malicious activities or events in Industrial Control Systems. It can be enforced using a predictive modeling method. This paper illustrates one such approach where LSTM and GRU are input a sequence of events and the predicted next step is displayed as output. The dataset is the event logs of an Energy Management System where *SCADA category* and *Device type* are taken as features and arranged into time series of 18 events. Two required labels *SCADA category* and *Device type* are output through two separate RNN models. With the help of *SCADA category*, the next SCADA event can be known, while *Device type* can be used to track down the device where the SCADA event will occur. High priority critical situations can be mitigated with this knowledge. A comparative analysis between LSTM and GRU for this predictive modeling application is demonstrated.

3.2 Dataset

Industrial Control Systems (ICS) cyber attacks dataset was provided on a google site by *uah.edu* (The University of Alabama in Huntsville). The dataset includes 30 days of events logged by an Energy Management System (EMS) at an investor-owned utility in the United States of America. Data were anonymized by changing the names of operators, devices, and facilities. There were 5758500 events in the data, arranged in rows, where each row was a unique event. Dataset consisted of several attributes like *EventId*, *Event Timestamp*, *SCADA category*, *TOC*, *AOR*, *Priority code*, *Substation*, *Device type*, *Device*, *event message*. Table 1 describes the dataset. The distribution of SCADA category events in the dataset according to Priority Code is depicted in Figure 1.

Table 1: Dataset description

Attributes	**Description**
EventId	A numerical value which is a count of events in the file.
Event Timestamp	Date and time of the event. The date is organized as *Year-Month-Day* where year is always 2017, the month is always 05 (May), and the day varies from 1-31.
SCADA category	The type of SCADA event being logged.
TOC	Indicates the source system.
AOR	Area of Responsibility, used to define the controlling authority (which operators).
Priority code	Code used to prioritize the events based on how critical they are. It ranges from 1 to 8, where 1 is the highest priority and 8 is the lowest
Substation	The name of the substation where the event originated.
Device type	The type of device from which the event originated.
Device	The Device column provides the name of the device which generated the event.
Event message	The event occurring is defined specifically.

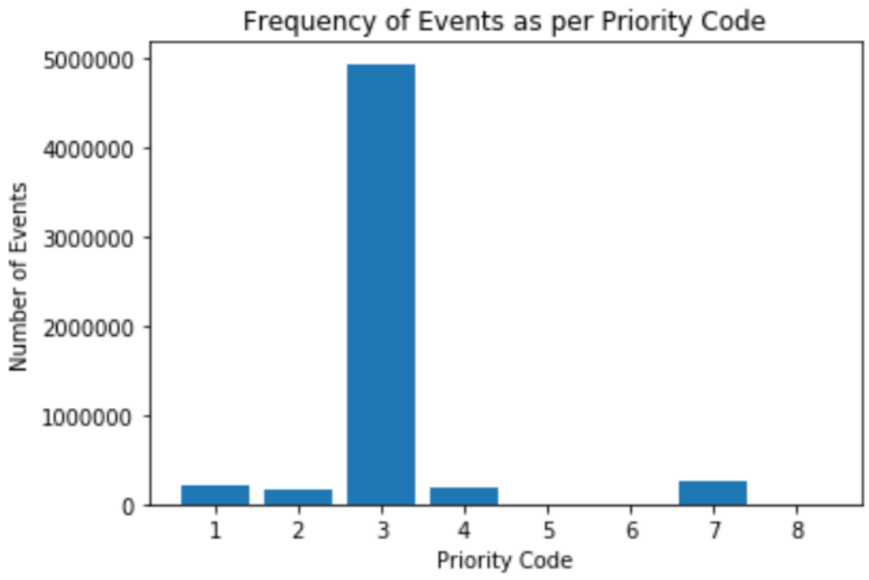


Figure 1: Distribution of Events to Priority Code

3.3 Preprocessing

- *Data Cleaning*
  The data contained inaccuracies and inconsistencies. Hence it was cleansed to remove these inaccurate and inconsistent records and irrelevant columns.

Table 2: Distribution of Devices by Device type

Statistical Measure	Number of Devices
mean	286.727
min	11.000
25%	39.750
median	108.000
75%	244.500
max	2763.000

- *Data Reduction*
  Data were analyzed statistically and the attribute subset selection technique was adopted to select highly relevant attributes. Each *SCADA category* has a *Priority Code* associated with it. Hence if the *SCADA category* of the next event is known, its criticality can be determined. Similarly the location of the next SCADA event can be pinpointed if the *Device* of the next SCADA event is predicted. Hence *SCADA category* and *Devices* were considered as highly relevant attributes. But the number of unique *Devices* was 11599. So they were clustered on the grounds of *Device type*. Smaller clusters of *Devices* having mean *Priority Code* greater than 7 were discarded. Table 2 shows the statistical values and distribution of *Devices* when grouped by *Device type*. This implies that if the *Device type* of the next SCADA event can be predicted, then, assuming *Normal distribution,* there is a 50% (median) probability that the number of devices of that particular device type is less than or equal to 108. It would be simple to monitor those 108 (approximately) devices to track the predicted SCADA event. Hence *Device type* was selected for the preprocessed dataset.

- *Data Transformation*
  One-hot encoding technique was employed on categorical variables to convert them to integer data. By this method, 47 *SCADA categories* and 44 *Device types* were one-hot encoded to form 91 inputs. The logs were then grouped into time series of length 18 to form features while the next event was taken as the label. In this manner, a dataset was constructed with one sample made up of:

  - Features: A time series of 18 events with each event consisting of 91 binary values (47 for *SCADA category* and 44 for *Device type*).
  - Labels: The *SCADA category* of the next event made up of 47 binary values for the

SCADA category model and the *Device type* of the next event made up of 44 binary values for the *Device type* model.

This preprocessed data was split into training and validation datasets in 75:25 ratio, employing *hold-out validation technique*. During each epoch the model was trained over and over again on the training data and it continued to learn about the features of data. In the testing phase, the model was used to evaluate and predict the next events on the reshuffled validation dataset.

3.4 Modeling

3.4.1 Long Short Term Memory (LSTM)

LSTM models are utilized to address the issue of vanishing gradients that is experienced in Recurrent Neural Network (RNN)(Hochreiter, S., and Schmidhuber, J., 1997). These networks are upgraded version of RNN that have been implemented for various sequence learning problems due to their scope in learning long-term dependencies (Nguyen, V. Q. et al., 2018). LSTMs are designed for applications where the input is an ordered sequence. In LSTM, the nodes are recurrent but they also have an internal state. The node uses an internal state as a working memory space which means information can be reserved and fetched over many timesteps. The input value, previous output, and the internal state are all used in the node's calculations. The results generated through computations provide an output value and update the current state. LSTM has parameters known as gates that control the flow of information within the node. These gate parameters are weights and biases, which means the behavior of the node depends on the inputs. Gates manipulate current information which is saved to the state and regulates output by the current calculation against saved information. So LSTM network is an exceptional type of RNN qualified for learning long-term dependencies. Figure 2 elaborates the structure of LSTM while its steps are given below:

- A *forget gate* layer identifies the information that is not mandatory and must be discarded from the cell state.

$$f_t = \sigma\left(W_f\left[h_{t-1}, x_t\right] + b_f\right) \tag{3.1}$$

- An *input gate* layer analyzes and decides the values to be updated. A hyperbolic tangent *(tanh)* layer then formulates candidate values, that would be appended to the cell state. This step determines the new information that will be stored in the cell state.

$$m_t = \sigma\left(W_m\left[h_{t-1}, x_t\right] + b_m\right) \tag{3.2}$$

$$c_{t1} = \tanh\left(W_c\left[h_{t-1}, x_t\right] + b_c\right) \tag{3.3}$$

- In this step, the previous cell state gets updated to the new cell state by forgetting the values which were decided to be unimportant previously and adding new candidate values to each state.

$$c_t = \left(f_t * c_{t-1}\right) + \left(m_t * c_{t1}\right) \tag{3.4}$$

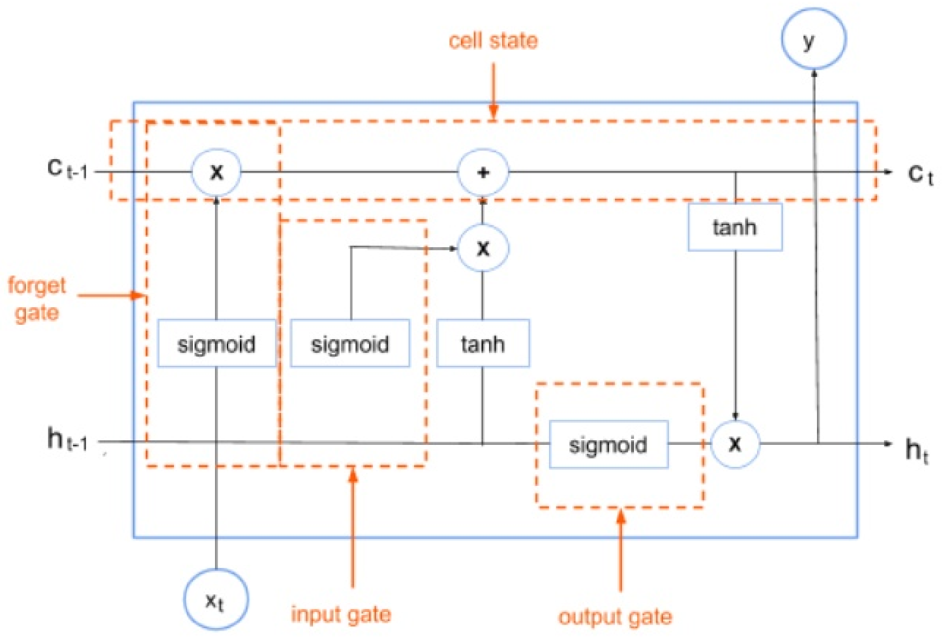


Figure 2: Long Short Term Memory

- In the final step, the *sigmoid* layer chooses the measures of the cell state to be declared as output by embedding a cell state through hyperbolic tangent *(tanh)* activation function and multiplying it by the *sigmoid* gate layer output.

$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right) \tag{3.5}$$

$$h_t = o_t * \tanh\left(c_t\right) \tag{3.6}$$

3.4.2 Gated Recurrent Unit (GRU)

It is a simpler variant of LSTM. Unlike LSTM, it consists of only three gates and does not maintain an Internal cell state unit. The gating units of GRU regulate the flow of knowledge inside the unit instead of storing it in a separate memory cell(Chung, J. et al., 2014). It merges the input and forget gates into an *update gate*. It also adds a "*reset gate*(Le, T. H. H. et al., 2019). Figure 3 explains this structure of GRU and its gates are described below:

- The *update gate* determines the amount of data from preceding timesteps that should be passed along to the next.

$$z_t = \sigma\left(W_z * x_t + U_z * h_{t-1}\right) \tag{3.7}$$

- The *reset gate* evaluates the extent of the data from preceding timesteps to forget.

$$r_t = \sigma\left(W_r * x_t + U_r * h_{t-1}\right) \tag{3.8}$$

- A new memory content will then store the pertinent data from the past, with the help of the *reset gate*.

$$h_{t1} = \tanh\left(W * x_t + r_t \odot \left(U * h_{t-1}\right)\right) \tag{3.9}$$

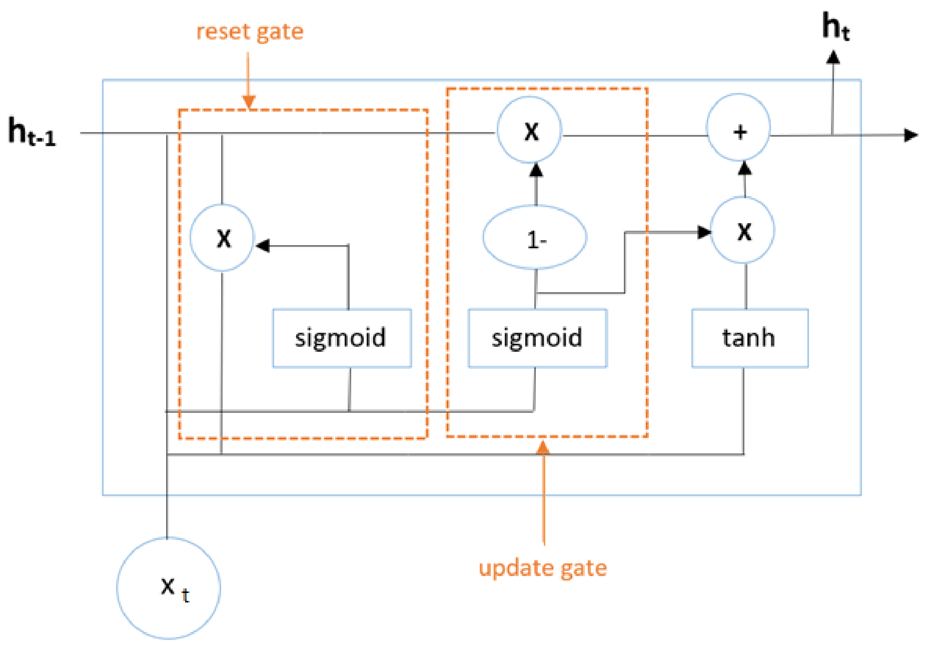


Figure 3: Gated Recurrent Unit

- The *update gate* determines the aggregation of data from the current step and that from previous state. For holding the current unit information, update gate calculates the output vector.

$$h_t = (z_t \odot h_{t-1} + (1 - z_t) \odot h_{t1}) \tag{3.10}$$

The weights(W) in both the models represent the strong connection between units and influence the gradient of the activation function and bias (b) is a constant that assists the model to better fit the data.

where h_{t-1}	= Output from the previous time step
x_t	= Input vector
h_t	= Output vector
c_t	= Cell state
c_{t1}	= New candidate value
z_t	= Update gate
r_t	= Reset gate
f_t	= Forget gate
m_t	= Input gate
o_t	= Output gate
h_{t1}	= Current memory content
σ	= Sigmoid function
$\tanh$	= Hyperbolic tangent

3.4.3 Similarities and differences between LSTM and GRU

RNNs undergo vanishing gradient problem, causing hindrance in learn the long data sequences. Both LSTM and GRU overcomes vanishing gradient problems which mainly occurs when parameters and hyperparameters are not set properly(Le, T. H. H. et al., 2019). GRU connects forget as well as input gates which is absent in LSTM. GRU employs less trainable parameters and therefore requires less memory and trains and executes quicker than LSTM. On the other hand, LSTM is performs better on datasets having longer sequences. If the sequence is large or accuracy is very critical, LSTM is preferred whereas, for less memory consumption and faster operation, GRU gets priority.

3.4.4 Stacked Predictive Models

In this paper, a stacked approach of executing RNN models is presented for prediction of *SCADA category* and *Device type* of next event. Two LSTM layers with 128 cells each along with an output layer having *softmax* activation function are implemented. As the input variables lead to multiclass classification, *categorical cross-entropy* loss function in conjunction with *Adam* optimizer is applied. Categorical cross-entropy loss function is used for single label categorization. Two such separate models are produced for the two outputs needed - *SCADA category* and *Device type*. GRU structures are designed in a similar fashion. Figure 4 and Figure 5 demonstrate the architectures of these models.

Softmax Activation Function calculation

The formula for calculation of softmax activation function is:

$$OutputLayer \longrightarrow \frac{e^{z_i}}{\sum\limits_{j=1}^{k}(e^{z_j})} \longrightarrow Probabilities \tag{3.11}$$

1. Every element of the output layer is passed through a standard exponential function and the results are summed up.
2. Each element of the output layer is exponentiated similarly, and then divided by the sum obtained in the step 1. This result is the probability of predicted class being the output *(multiclass classification problem).*

Categorical Cross-entropy Loss calculation

The mathematical formula for Categorical cross-entropy is:

$$Loss = \frac{-1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} 1_{y_i \in C_c} \log(p_{model}[y_i \in C_c]) \tag{3.12}$$

$N =$ number of Observations.

$1_{y_i \in C_c} =$ binary(boolean) function that indicates whether the *ith* observation belongs to the *cth* category.

$p_{model}[y_i \in C_c] =$ probability predicted by the model for the *ith* observation to belong to the *cth* category *(softmax activation function).*

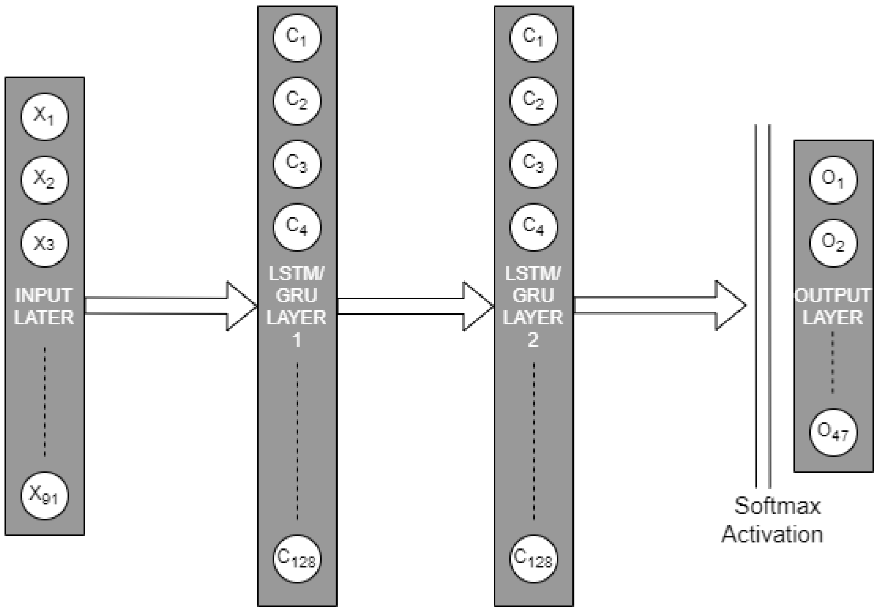


Figure 4: SCADA category predictive model

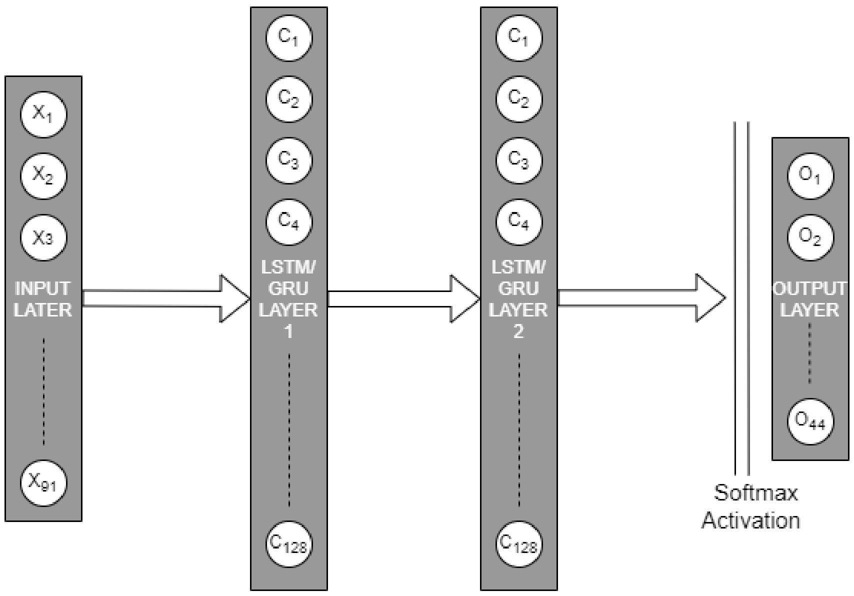


Figure 5: Device type predictive model

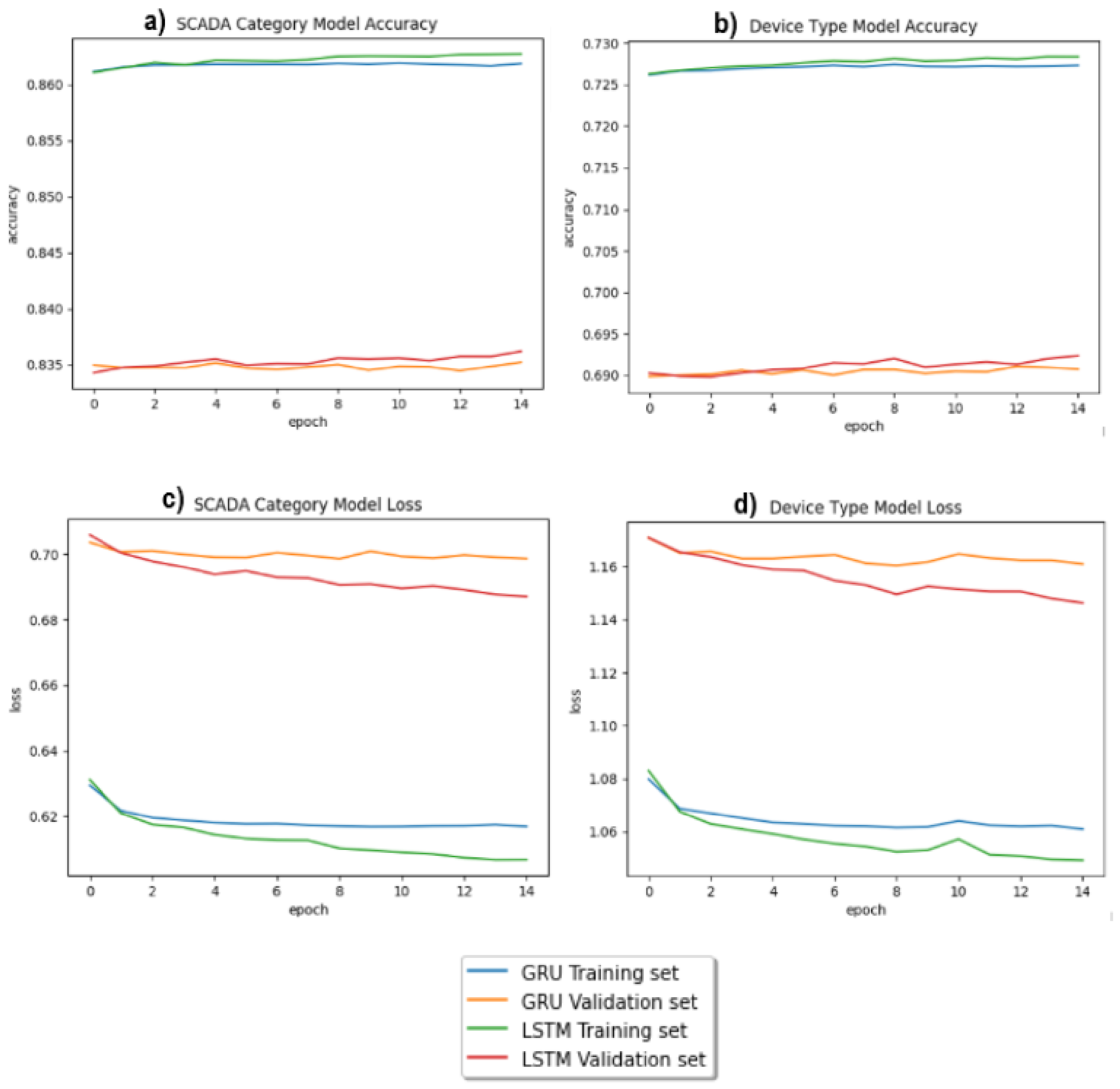


Figure 6: Line graph of (a) SCADA Category Model Accuracy (b) Device Type Model Accuracy (c) SCADA Category Model Loss (d) Device Type Model Loss

4 Results and Discussions

4.1 Training Result

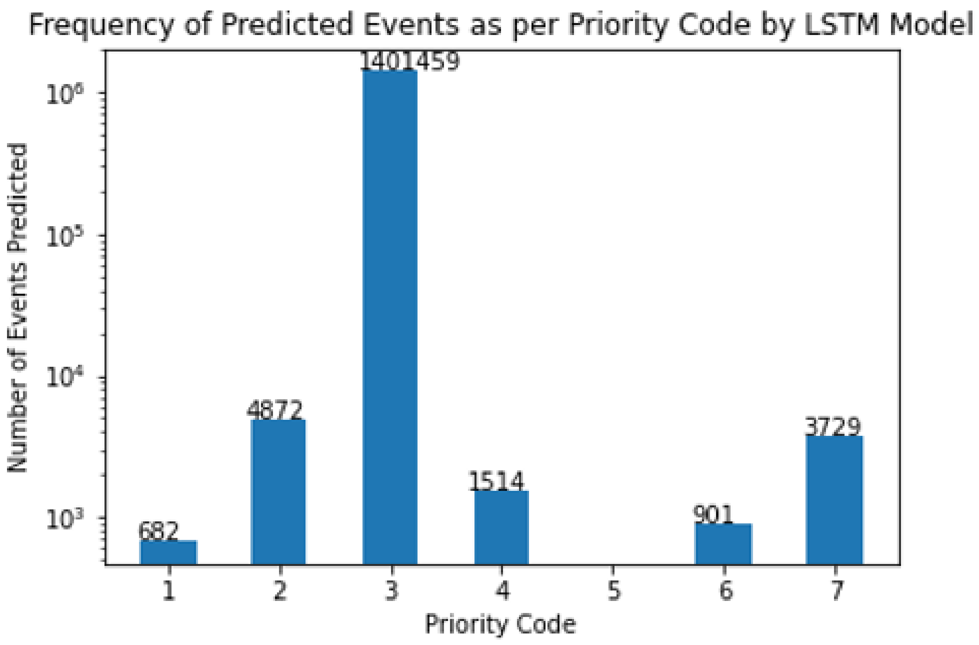
Four models were designed - LSTM for *SCADA category*, GRU for *SCADA category*, LSTM for *Device type* and GRU for *Device type*. They were trained with time series of 18 events, for 15 epochs. This training and validation phase was done on DGX Station on 4 Graphics Processing Units (GPU) and is portrayed above. The accuracies and losses of the 4 models are plotted in Figure 6.

4.2 Hyperparameters related to Network structure and Training Algorithms

Hyperparameters deal with higher level abstractions and decisions about the model such as its complexity, capacity to learn, etc. They can be fixed before training or determined after trial and error. *Random search* method is used to tune the hyperparamaters. As the name suggests,it is a technique where random combinations of the hyperparameters are considered in every iteration to find the best solution to build the model.

- *Optimizer hyperparameters :* They are related more to the optimization and training process.
  - *Learning rate :* Learning rate controls the length of the stride that the model takes during gradient descent and hence the time required to train the model. If learning rate is too large, the weights oscillate around the optimum without reaching it. If its too small, the weights take too long to reach optimum. A learning rate of 0.001 was used for this model.
  - *Batch size :* The batch size determines the number of samples that will pass through the model in one cycle before backpropagation while implementing *mini-batch gradient descent.* For this learning algorithm, batch sizes generally used are 32, 64, and 128 samples, of which batch size 64 is employed in this paper.
  - *Number of epochs :* An epoch refers to one cycle of the model through the entire training dataset. The model is trained on the training dataset multiple times to to get as accurate a model as possible. However, if the number of epochs is too large, it leads to overfitt the training data. Hence, taking the size of the training dataset into account, the model was trained on 15 epochs.
  - *Optimizer :* Adam is the optimizer derived from adaptive moment estimation and employed in this paper.It generally performs the best in many types of applications, models and datasets.
  - *Activation Function :* Softmax activation function is employed in the final output layer to solve the multiclass classification problem. Softmax is often used to normalize the output of a network to fit a probability distribution over predicted output classes. Hyperbolic tangent *(tanh)* activation function is applied as activation function for the LSTM layers.
- *Model-specific hyperparameters :* They are more involved in the structure of the model.
  - *Number of hidden layers :* 2 LSTM/GRU layers are stacked in this model where the hidden sequence output of the preceding LSTM/GRU layer is input to the next LSTM/GRU layer. In stacked LSTM/GRU model, the additional hidden layer is added to a neural network , albeit at the cost of more training time, to make it deeper, recombining the learned representations from the preceding layer to create new representations in the next layer at higher levels of abstraction.
  - *Number of hidden cells :* The number of hidden units decides the number of trainable parameters in the model and ability to model complexity relationships. One rule-of-thumb method for deciding the number of neurons in the hidden layers is that, the number of neurons should be less than twice the number of inputs to that layer. Ultimately, the selection of an architecture for the model comes down to trial and error. Hence, the model in this paper has 128 hidden cells in each LSTM/GRU layer.

a)



b)

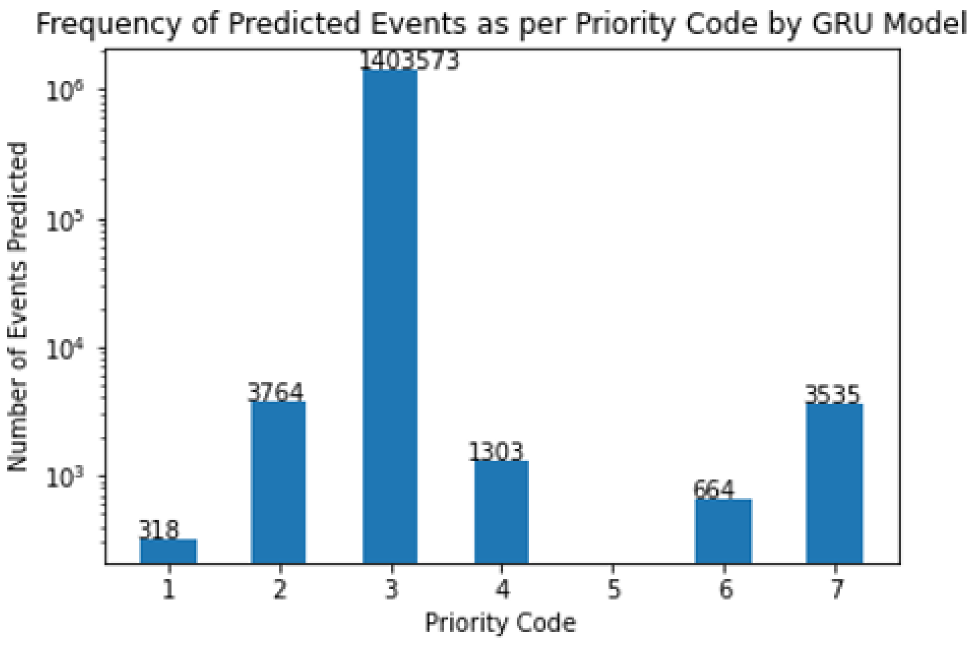


Figure 7: Distribution of Predicted Events to Priority Code in (a)LSTM and (b)GRU Models

Table 3: Comparison between Models

	LSTM-based SCADA category predictive model	**GRU-based SCADA category predictive model**	**LSTM-based Device type predictive model**	**GRU-based Device type predictive model**
Memory	45,56,954 bytes	36,09,296 bytes	45,54,789 bytes	36,06,012 bytes
Testing Time	15 ms/step	13 ms/step	15 ms/step	13 ms/step
Accuracy	0.8129	0.8138	0.6609	0.6620
Loss	0.7624	0.7540	1.2320	1.2208

4.3 Model evaluation

Each *SCADA category* has a *Priority code* associated with it. Hence the *Priority code* of the next SCADA event can be found from the predicted *SCADA category*. If a high priority SCADA event is predicted, the operator can be informed of it. In case the SCADA event is anomalous or identified as a threat, actions can be taken to mitigate it immediately. The plots in Figure 7 denote the distribution of *Priority codes* in the predictions made by the models during testing phase.

4.4 Comparative analysis

During the testing phase, the models were evaluated on a reshuffled validation dataset and compared on the grounds of the result. LSTM gives better accuracy and loss than GRU over the validation phase. On the other hand, GRU marginally outperforms LSTM during testing. Besides that, GRU requires less space and trains faster than LSTM. Further, GRU predicts the results more rapidly, which can be very advantageous for real-time applications. Table 3 consolidates these results of the testing phase.

5 Conclusion

This paper inspects the effectiveness of using stacked LSTM and GRU in designing data-driven predictive models for anomaly detection, on a dataset of Energy Management System logs by exploiting the relationship between data elements in the logs and the time series structure of communication patterns between devices in the network. A comparison between LSTM and GRU models is demonstrated. While the performances of GRU model is comparable to that of LSTM in terms of accuracy and loss, GRU requires significantly less space and is considerably faster than LSTM.

Acknowledgement

This paper is supported by CoE-CNDS, VJTI which maintained and gave access to the infrastructure required for the implementation of this project. We would also like to thank Dr. Faruk Kazi (Dean-Research, Development and Consultancy at VJTI) and faculty, which provided invaluable expertise and insights regarding the paper.

References

Allhoff, F., and Henschke, A. 2018. The Internet of Things: Foundational ethical issues, *Internet of Things* **1-2**, 55–66.

Amarasinghe, K., Marino, D. L., and Manic, M. 2017. Deep neural networks for energy load forecasting, *IEEE 26th International Symposium on Industrial Electronics (ISIE)* pp. 1483–1488.

Ashok, A., Hahn, A., and Govindarasu, M. 2014. Cyber-Physical Security of Wide-Area Monitoring, Protection and Control in a Smart Grid Environment, *Journal of Advanced Research* **5-4**, 481–489.

Chapman, J. P., Ofner, S., and Pauksztelo, P. 2016. Key Factors in Industrial Control System Security, *IEEE 41st Conference on Local Computer Networks (LCN)* pp. 551–554.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling, *arXiv preprint arXiv:1412.3555* .

Dumitru, C. D., and Gligor, A. 2012. SCADA based software for renewable energy management system, *Procedia Economics and Finance* **3**, 262–267.

Feng, C., Li, T., and Chana, D. 2017. Multi-level Anomaly Detection in Industrial Control Systems via Package Signatures and LSTM Networks, *47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)* pp. 261–272.

Gao, J., Gan, L., Buschendorf, F., Zhang, L., Liu, H., Li, P., Dong, X., and Lu, T. 2019. LSTM for SCADA Intrusion Detection, *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)* pp. 1–5.

Ghanem, T. F., Elkilani, W. S., and Abdul-kaderc, H. M. 2015. A Hybrid Approach for Efficient Anomaly Detection using Metaheuristic Methods, *Journal of Advanced Research* **6-4**, 609–619.

Hasan, M., Islam, M. M., Zarif, M. I. I., and Hashem, M. M. A. 2019. Attack and Anomaly Detection in IoT Sensors in IoT Sites Using Machine Learning Approaches, *Internet of Things* **7**, 100059.

Hinkka, M., Lehto, T., and Heljanko, K. 2019. Exploiting Event Log Event Attributes in RNN Based Prediction, *New Trends in Databases and Information Systems* pp. 405–416.

Hochreiter, S., and Schmidhuber, J. 1997. Long Short-term Memory, *Neural Computation* **9-8**, 1735–1780.

Hollingsworth, K., Rouse, K., Cho, J., Harris, A., Sartipi, M., Sozer, S., and Enevoldson, B. 2018. Energy Anomaly Detection with Forecasting and Deep Learning, *2018 IEEE International Conference on Big Data (Big Data)* pp. 4921–4925.

Iqbal, R., Maniak, T., Doctor, F., and Karyotis, C. 2019. Fault Detection and Isolation in Industrial Processes Using Deep Learning Approaches, *IEEE Transactions on Industrial Informatics* **15**(5), 3077–3084.

Kargl, F., van der Heijden, R. W., König, H., Valdes, A., and Dacier, M. C. 2014. Insights on the Security and Dependability of Industrial Control Systems, *IEEE Security Privacy* **12**(6), 75–78.

Khodayar, M., Kaynak, O., and Khodayar, M. E. 2017. Rough deep neural architecture for short-term wind speed forecasting, *IEEE Transactions on Industrial Informatics* **13**(6), 2770–2779.

Kim, Y., and Kim, Y.-S 2017. Optimizing neural network to develop loitering detection scheme for intelligent video surveillance systems *International Journal of Artificial Intelligence* **15**, 30–39.

Le, T. H. H., Kim, Y., and Kim, H. 2019. Network Intrusion Detection Based on Novel Feature Selection Model and Various Recurrent Neural Networks, *Applied Sciences* **9**, 1392.

LeCun, Y., Bengio, Y., and Hinton, G. 2015. Deep learning, *Nature* **521**, pp. 436–444.

Mesaric, P., and Palasek, B. 2014. Supervisory control and data acquisition for energy management systems, *Proceedings of 2014 International Scientific and Professional Conference Contemporary Issues in Economy and Technology, CIET* .

Montáns, F., Chinesta, F., Gómez-Bombarelli, R., and NathanKutz, J. 2019. Data-driven modeling and learning in science and engineering, *Comptes Rendus Mécanique, Data-Based Engineering Science and Technology* **347**(11), 845–855.

Murray, G., Johnstone, M. N., and Valli, C. 2017. The convergence of IT and OT in critical infrastructure, *The Proceedings of 15th Australian Information Security Management Conference* pp. 149–155.

Nguyen, V. Q., Ma, L. V., and Kim, J. 2018. LSTM-based anomaly detection on big data for smart factory monitoring, *Journal of Digital Contents Society* **19-4**, 789–799.

Nino-Ruiz, E., Ardila, C., and Capacho, R. 2018. Local search methods for the solution of implicit inverse problems, *Soft Computing* **22**, 4819–4832.

Nino-Ruiz, E., and Yang, X. 2019. Improved tabu search and simulated annealing methods for nonlinear data assimilation, *Applied Soft Computing* **83**, 105624.

Oprea, M. 2012. Intellenvq-air: An intelligent system for air quality analysis in urban regions *International Journal of Artificial Intelligence* **9**, 106–122.

Precup, R., and Hellendoorn, H. 2011. A survey on industrial applications of fuzzy control, *Computers in Industry* **62**(3), 213–226.

Precup, R., and Tomescu, M. 2015. Stable fuzzy logic control of a general class of chaotic systems, *Neural Computing and Applications* **26**(3), 541–550.

Sayed, K., and Gabbar, H.A. 2017. SCADA and smart energy grid control automation, *Smart Energy Grid Engineering* pp. 481–514.

Shi, Z., Liang, H., and Dinavahi, V. 2018. Direct interval forecast of uncertain wind power based on recurrent neural networks, *IEEE Transactions on Sustainable Energy* **9**(3), 1177–1187.

Skripcak, T., and Tanuska, P. 2013. Utilisation of on-line machine learning for SCADA system alarms forecasting, *Proceedings of 2013 Science and Information Conference, SAI* pp. 477–484.

Stouffer, K., Lightman, S., Pillitteri, V., Abrams, M., and Hahn, A. 2015. Guide to Supervisory Control and Data Acquisition (SCADA) and Industrial Control Systems Security, *Recommendations of the National Institute of Standards and Technology, Special Publication 800-82* .

Teixeira, A., Dán, G., Sandberg, H., and Johansson, K.H. 2011. A Cyber Security Study of a SCADA Energy Management System: Stealthy Deception Attacks on the State Estimator, *IFAC Proceedings Volumes* **44**, 11271–11277.

Venayagamoorthy, G. K., Sharma, R. K., Gautam, P. K., and Ahmadi, A. 2016. Dynamic energy management system for a smart microgrid, *IEEE Transactions on Neural Networks and Learning Systems* **27**(8), 1643–1656.

Wang, H. and Lei, Z. and Zhang, X. and Zhou, B., and Peng, J. 2019. A review of deep learning for renewable energy forecasting, *Energy Conversion and Management* **198**, 111799.

Xiao, P., Venayagamoorthy, G. K., Corzine, K. A., and Huang, J. 2010. Recurrent Neural Networks Based Impedance Measurement Technique for Power Electronic Systems, *IEEE Transactions on Power Electronics* **25**(2), 382–390.