IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 4 August 2023, accepted 25 August 2023, date of publication 1 September 2023, date of current version 8 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3311393

RESEARCH ARTICLE

Detection and Localization of Data Forgery Attacks in Automatic Generation Control

FENGLI ZHANG, YATISH DUBASI, WEI BAO, AND QINGHUA LI, (Member, IEEE)
Department of Electrical Engineering and Computer Science, University of Arkansas, Fayetteville, AR 72701, USA

Corresponding author: Qinghua Li (qinghual@uark.edu)

This material is based upon work supported by the Department of Energy under Award DE-OE0000779.

ABSTRACT Automatic Generation Control (AGC) is a key control system to maintain the power system's balance between load and supply by maintaining its frequency in a specific range. It collects the tie-line power flow and frequency measurements of each control area to calculate the Area Control Error (ACE) and then adjusts power generation based on the calculated ACE. However, malicious frequency or tie-line power flow measurements can be injected and then AGC is misled to make false power generation adjustments which will harm power system operations. Such attacks can be carefully designed to pass the power system's existing bad data detection schemes. In this work, we propose Long Short Term Memory (LSTM) neural network-based methods and a Fourier Transform-based method to detect and localize such data forgery attacks in AGC. These methods only utilize historical data, which are already available in existing AGC systems, making them easy to deploy in the real world. They learn normal data patterns from historical data and detect abnormal patterns caused by attacks. To make it easier for users to use the solution, we also propose methods to automatically find the proper detection threshold based on user needs. These methods are tested both on real and simulated datasets and show high detection and localization accuracy.

INDEX TERMS Power grid, automatic generation control, data forgery, deep learning, attack detection, attack localization.

I. INTRODUCTION

Automatic Generation Control (AGC) is a key control system in the power grid which aims to keep balance between power generation and load and maintain a stable power system frequency. It automatically adjusts the power generation in response to the area control imbalance. A power system usually consists of multiple interconnected control areas. A control area is connected to its neighboring areas through tie-lines, and power sharing between two neighboring areas occurs on these tie-lines. Each control area has its own AGC with the function to adjust the amount of power generation to keep its frequency in the scheduled range and keep the power exchanges with other areas to the values agreed upon in economic dispatch. The required adjustment in power generation in each control area is called Area Control Error (ACE), which is calculated based on the frequency deviation and power exchange deviation. ACE is calculated every 2 to 4 seconds, and then the set-point of the generator governors participating in AGC is changed based on the calculated ACE.

In AGC, a control center periodically collects the power system's frequency and tie-line power flow measurements from meters to calculate ACE. However, the increasing number of smart devices and connections to external networks in the power grid make power systems more vulnerable to cyber attacks, such as data forgery attacks. Existing methods implemented in control centers, such as State Estimation (SE) and Bad Data Detection (BDD) [1], are unable to detect intelligent cyber attacks nor guarantee the reliability of the power system. The work in [2] shows that existing schemes can be bypassed by carefully designed data forgery attacks.

If malicious tie-line power flow or frequency data is injected into normal measurements to mislead AGC into making miscalculations, AGC may issue wrong commands based on the miscalculated ACEs [3]. As an example, when ACE is positive, implying the area is over generating power, AGC should decrease the power generation. However, if false

The associate editor coordinating the review of this manuscript and approving it for publication was Giambattista Gruosso.

tie-line power flow or frequency data is injected and results in a negative ACE value, AGC will believe that the area is under generating and issue a command to increase the power generation, which exaggerates the over-generating situation. If these attacks go undetected, they can lead to physical damage to consumer electronics or even damage to power grid equipment. Such attacks can also induce power grid responses such as disconnecting generators or customer load. Additionally, this can further lead to cascading failures and large-scale blackouts. Attackers already have the resources and capabilities to conduct sophisticated attacks against the power grid to cause power outages [4], [5], [6]. AGC is a particularly attractive target when attacking the power grid because it is a highly automated system with minimal human supervision or intervention. Therefore, once an AGC system is compromised, the impact of the attack can escalate very quickly.

In this work, we propose Neural Network-based and Fourier Transform-based approaches to detect and localize data forgery attacks in AGC. In the first approach, we adopt a Long Short Term Memory (LSTM) neural network to learn ACE patterns from historical data, predict the ACE sequence pattern for the next detection window based on the learned patterns, and finally compare the predictions with the corresponding ACE sequence pattern calculated from measurements to determine whether there is forged data in the measurements. The LSTM model can be built with a single feature: the historical ACE data (i.e., *single-feature LSTM*). It can also include more related features (i.e., *multi-feature LSTM*), such as frequency and tie-line power flow, to achieve better performance. The LSTM-based approach is also developed to localize attacks by checking which measurement is abnormal, to know which meter is compromised. In the second approach, we convert ACE and measurement data from the time domain to the frequency domain by using Fourier Transform and then check if ACE and measurement data are normal in the frequency domain. To make it easier for users to use the solutions, we also propose methods for automatically finding the appropriate detection threshold based on user needs. We test these approaches on both synthetic and real datasets. Our detection methods can be easily deployed since we only use historical data that are already available in current AGC systems.

This paper's main contributions are summarized as follows.

- We study the use of single-feature and multi-feature LSTM methods to detect and localize data forgery attacks in AGC. LSTM's great prediction accuracy on ACE and measurement data allows these methods to have very high attack detection and localization rates.
- We study the use of Fourier Transform methods to detect and localize data forgery attacks in AGC. It was observed that the ACE and measurement data vary significantly in the frequency domain if false data was injected. Therefore, this method also has high attack detection and localization rates.
- We present a method to automatically generate a proper detection threshold for our designed attack detection methods. That makes it easier for a user to use our solution.
- We thoroughly evaluate the effectiveness of our designed methods using synthetic and real world data, and compare them with an existing solution.

The remainder of this paper is organized as follows. We discuss related work in Section II. Section III describes the basics of AGC systems and attack models used in this work. The proposed detection and localization methods are presented in Section IV. Section V describes the simulated system and the simulated and real datasets. Section VI shows the performance of the proposed methods on both simulated and real data. We conclude our work in Section VII.

II. RELATED WORK

Much work has been done in power grid security [7], [8]. Here, we mainly review related work on attacks in AGC.

The work in [9] developed a model-based method that predicts ACE values every 5 minutes and then compares the predictions with real ACEs to detect anomalies. However, the predictions depend heavily on the load forecast, which might be affected by any errors in the load forecast. In [10], the authors proposed a two-tier intrusion detection system, which first predicts the next ACE value based on the current ACE value and then passes the ACE to the second tier to verify whether the ACE deviates from its prediction. However, that work only used one time point's ACE for prediction and is prone to error.

The work in [11] utilizes a Kalman filter to predict signal states or actual measurements. Next, the predicted values are sent to a χ^2 detector in order to detect attacks. Another work [12] employs a two-stage Kalman filter to simultaneously estimate attack-free and potentially forged measurements. This allows users to perform attack detection. However, Kalman filter-based approaches are sensitive to noise or disturbances in measurements.

The work in [13] proposed an approach to predict how much false data was potentially added from a false data injection attack. However, it only looked at attacks to the frequency measurement, and it only considered two attack types: constant or periodic false data injection. Another work [14] developed a solution to predict what the actual signal measurement would be if the signal were forged. A limitation of this work is that it considers a threat model where the attacker is only able to attack one signal at a given time. Therefore, the proposed method will not work well when multiple signals are attacked. Moreover, only ramp attacks and pulse attacks were considered. The work in [15] utilized an autoencoder network to reconstruct input sample sequences with minimum reconstruction residual and detect attacks based on it. The nature of this autoencoder approach makes it easy to overfit the model in such a way that attacks that could agree with the correlation of the original system cannot be detected. These works mainly focus on recovering

the original signal measurement. Different from them, our work focuses on detection and localization of data forgeries.

The work in [16] proposes an ensemble method consisting of K-means clustering and support vector machines (SVMs) to be applied to area-level features to detect attacks in a decentralized manner. This method requires the installation of this software on all local generators in the grid.

The work in [17] and [18] utilize fuzzy theory with machine learning methods such as support vector machines or deep neural networks with LSTM to detect attacks. These methods consider the impact of various false data injection attacks to classify the characteristics of the attacks. These methods are rigid and can be weak against hybrid attacks or attack types that were not included in the training.

The work in [3] explored how to launch effective attacks in the shortest time, but it did not provide any detection method. Another work [19] presented how to determine the best response strategies against attacks based on a security game model. Our previous work [20] proposed a physics-based method to detect and localize data forgery attacks in AGC. The method utilizes an alternative way of computing ACE by using a more detailed power system model of the control area, but it requires more noise-free measurements and more complex computations. A deep learning-based method, Conditional Deep Belief Network, was developed to detect data forgery attacks [21]. It requires attacked data to be the training dataset, and thus the model performance heavily depends on what types of attacks are included in the training dataset. If an attack type is not included in the training dataset, it will be difficult to detect. Additionally, there are not many available attacked data in practice that can be used as training data.

The preliminary versions of this work [20], [22] studied using a single-feature LSTM model for attack detection/localization and using a Fourier Transform based method for attack detection, but they did not consider multi-feature LSTM nor Fourier Transform based attack localization. This paper also extends the author's dissertation [23] by proposing methods for automatic generation of detection thresholds for the two detection methods and evaluating their effectiveness.

III. PRELIMINARY

A. AGC SYSTEM

Each control area has an AGC to keep its frequency in a scheduled value by adjusting power generation. AGC is an automated control system and its workflow is illustrated in Fig. 1. It periodically (every 2×4 seconds) receives frequency measurements from the control area, receives tie-line power flow measurements between this area and neighboring areas from field devices, and then calculates the ACE according to the equation:

$$ACE = (P_{tieline} - P_{sch}) + B(f - f_{sch}) \tag{1}$$

where P_{sch} and f_{sch} are the scheduled tie-line power flow and the scheduled frequency respectively. $P_{tielnie}$ and f are measurements. B is the frequency bias factor, which is constant for each power system and is estimated annually. Then AGC adjusts the power generation of generators according to the calculated ACE.

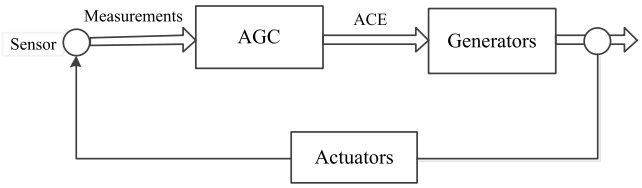


FIGURE 1. AGC system.

B. ATTACK MODELS

To cause miscalculations of ACEs, attackers can inject false data to frequency measurements or tie-line power flow measurements. ACE is within a specific range under normal conditions. Once ACE exceeds the limit, an alarm will be triggered to human operators and the AGC may be suspended [24]. If an attacker injects a significant attack into the measurements, such as tie-line power flow, it might trigger the alarm. Thus, the attacker might only modify the measurements slightly so that the attacks do not trigger the alarm. However, a single marginally forged data measurement may not be enough to introduce a significant impact on the power grid. According to [3], the shortest time to stealthily mislead the system frequency to breach the safety condition without triggering AGC suspension is at least 10 AGC cycles, which means that in order to achieve expected effects, the attacker needs to inject a series of false data to indirectly control the generator for a period of time.

One way to launch stealthy attacks is to find the maximum or minimum values of normal frequency or tie-line power flow measurements, and then inject the minimum or maximum data values as measurements. Since the maximum and minimum values are still within the normal range, they will be accepted by AGC. Let T_a represent the attack period, t represent time, t_0 represent the time point when the attacker starts an attack, $y(t)$ represent the true measurement (which could be either frequency or tie-line power flow as discussed later) value without attacks, and $y^*(t)$ represent the measurement value with possible attacks.

1) MAX ATTACK

This attack replaces the measurements with the maximum data value y^{max}.

$$y^*(t) = \begin{cases} y(t), & \text{if } t \notin T_a \\ y^{max}, & \text{if } t \in T_a \end{cases} \tag{2}$$

2) MIN ATTACK

This attack replaces the measurements with the minimum data value y^{min}.

$$y^*(t) = \begin{cases} y(t), & \text{if } t \notin T_a \\ y^{min}, & \text{if } t \in T_a \end{cases} \tag{3}$$

In addition to the Max and Min attacks, we also consider three attack models which are also explored in the literature [9], [25]: scaling attack, ramp attack, and random attack. In these attack models, the attacker keeps launching attacks until they achieve the expected results. The three attack models are described as follows.

3) SCALE ATTACK

The attack scales measurement values up or down by a scaling parameter λ_s.

$$y^*(t) = \begin{cases} y(t), & \text{if } t \notin T_a \\ y(t) + \lambda_s \cdot y(t), & \text{if } t \in T_a \end{cases} \tag{4}$$

4) RAMP ATTACK

The attack gradually modifies measurements by adding $\lambda_r(t - t_0 + 1)$. λ_r is a ramping parameter. This type of attack could be more difficult to detect because it has very small and unnoticeable changes at the beginning of the attack period.

$$y^*(t) = \begin{cases} y(t), & \text{if } t \notin T_a \\ y(t) + \lambda_r(t - t_0 + 1), & \text{if } t \in T_a \end{cases} \tag{5}$$

5) RANDOM ATTACK

The attack modifies measurement values by adding some random values in a range with lower bound λ_a and upper bound λ_b during the attack period.

$$y^*(t) = \begin{cases} y(t), & \text{if } t \notin T_a \\ y(t) + rand(\lambda_a, \lambda_b), & \text{if } t \in T_a \end{cases} \tag{6}$$

For the same power system, the higher the attack parameters λ are, the more significant the attacks are.

IV. DETECTION METHODS

The ACE time series of a control area has its specific patterns determined by the system's physical configurations. Fig. 4 shows the pattern of 250 cycles' ACE data from a real dataset denoted by *Real1* [26]. If an attack is injected, the pattern will be changed. Thus, attacks can be detected by checking whether there are ACE data patterns that deviate from normal patterns. Following this idea, we can learn normal patterns of ACEs with a neural network and check whether new ACEs match the normal patterns. The detection method overview is shown in Fig. 2.

A. LSTM NEURAL NETWORK

To determine whether new ACEs' pattern is normal, the neural network model should be able to find which pattern it resembles and link the current observations with the past ones. Recurrent Neural Network (RNN) is designed to deal with data with dependency [27], whose structure is shown in Fig. 3. RNN can be regarded as multiple copies of the same neural network which are connected successively. The output of a neural network will be passed to its successor. X_{t-1} is the input data and h_{t-1} is the output data at time point $t-1$. h_{t-1} will be passed to the next neural network. It together with X_t will be the input for the next neural network. The output is calculated as: $h_t = \sigma(W_h X_t + U_h h_{t-1} + b)$, where W_h, U_h, b are the parameters and σ is the activation function in the neural network. In this way, the RNN model allows the information to be passed and persist.

However, standard RNN is not good at addressing long-term dependency. As the time sequence moves forward, the previous information carried by the neural network will become less and less, and eventually vanish. However, in our problem, we need to find what patterns the current observations follow, which requires the neural network to

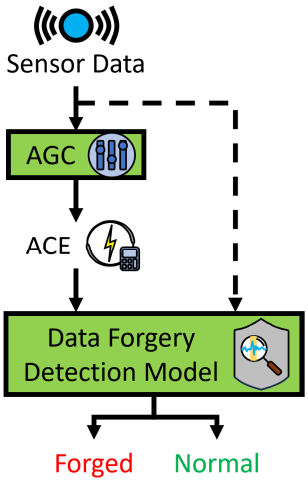


FIGURE 2. Solution overview.

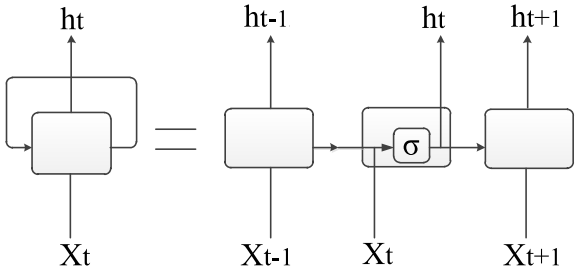


FIGURE 3. Structure of recurrent neural network.

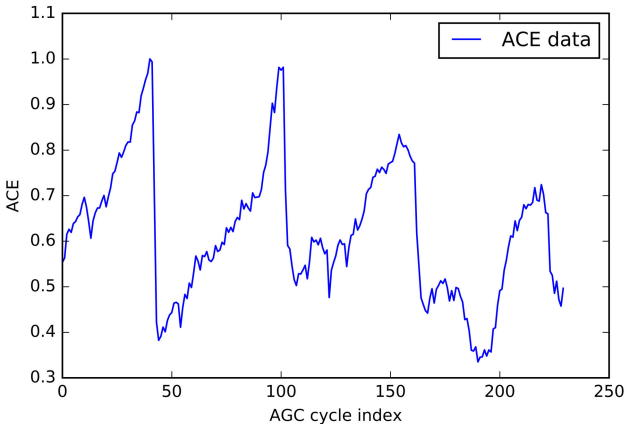


FIGURE 4. ACE data pattern.

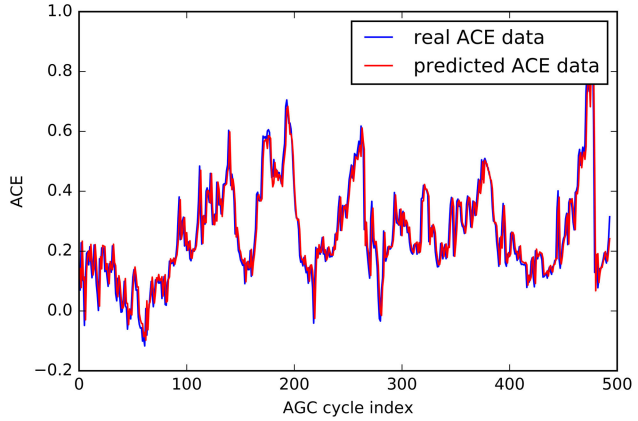


FIGURE 5. Prediction with LSTM.

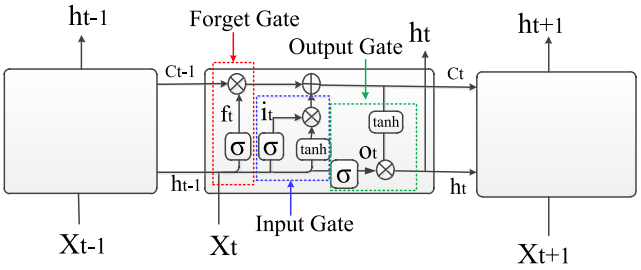


FIGURE 6. Structure of long short term memory network.

remember and relate to previous patterns even far from the current time point. In order to solve the long-term dependency problem, a type of RNN named Long Short Term Memory (LSTM) [28] is adopted, whose structure is shown in Fig. 6.

In LSTM, 'forget gate' f_t and 'input gate' i_t are added in each neural network chunk. The forget gate is to decide which information needs to be discarded. The input gate is to decide which new information is going to be stored. Thus, these gates help remove useless information while remembering useful information for a long period of time. As shown in Fig. 6, not only h_{t-1} is passed to the next neural network, but also C_{t-1}. C_t is a memory cell to store remembered information. The output h_t and C_t are calculated as follows:

$$\begin{aligned} f_t &= \sigma(W_f X_t + U_f h_{t-1} + b_f) \\ i_t &= \sigma(W_i X_t + U_i h_{t-1} + b_i) \\ C' &= tanh(W_c X_t + U_c h_{t-1} + b_c) \\ O_t &= \sigma(W_o X_t + U_o h_{t-1} + b_o) \\ C_t &= C_{t-1} \cdot f_t + i_t \cdot C' \\ h_t &= O_t \cdot tanh(C_t) \end{aligned} \tag{7}$$

where W, U, b are the neural network weight parameters. The important information will flow through the memory cell C_t. For our problem, LSTM is able to remember and find which previous sequence pattern the current sequence resembles or is similar to. Then it can predict the next sequence pattern based on the resembled pattern.

B. LSTM-BASED DETECTION METHOD

In order to accurately detect attacked data through comparison with predictions, the LSTM model should make accurate predictions. We test the LSTM model's prediction performance on the *Real1* dataset that only contains real ACEs [26]. Here, we build a model that predicts the next ACE value based on the previous five ACE values. The first 67% of the *Real1* dataset is used as the training dataset to train the model to learn ACE patterns, and the latter 33% of the dataset is used to test the trained model's prediction accuracy, which is shown in Fig. 5. The blue line is the real ACE values and the red line is the predictions that fit real data very well. The results show that the LSTM model can achieve high prediction accuracy.

The high accuracy in Fig. 5 is obtained when only ACE data is used as the feature. The LSTM model's performance might be further improved by considering more relevant features. Specifically, the *ACE* value depends on frequency and tie-line power flow measurement according to Eq. 1, while the frequency and tie-line power flow are affected by the variance between real load and power generation. Thus, ACE values are closely related to frequency, tie-line power flow and real load. We consider these three factors in addition to the past *ACE*. The input of each time point will be $X_t = \{f_t, P_t, R_t, ACE_t\}$, where f_t, P_t, R_t, ACE_t are the frequency, tie-line power flow, real load, and ACE at time point t respectively.

In particular, the LSTM model should be first trained and tuned with historical dataset and can be updated dynamically (e.g., every year) to include newly generated data. For attack detection, the recent data sequence is fed into the trained model to predict the next time point's value. The length of the input sequence can be determined or adjusted based on different AGC systems to get better performances. Let m denote the input sequence length. X_t is the data value at time point t. X_t can be a single feature $\{ACE_t\}$ (*single-feature LSTM*) or multiple features $\{f_t, P_t, R_t, ACE_t\}$ (*multi-feature LSTM*). We use the input data sequence $(X_{(t+1)}, X_{(t+2)}, \ldots, X_{(t+m)})$ to predict $X_{(t+m+1)}$. When a predicted data sequence with n data points is available, the n predicted ACE values are compared with the calculated ACE values at the same time points to check whether the calculated ACE data sequence deviates significantly from the predicted. If so, it is detected as an attacked sequence. The detailed steps of the method are shown as follows.

Step 1: Predict the next data sequence with the trained model.

$$\begin{aligned} X_{t-m+1}, X_{t-m+2}, \ldots, X_t &\rightarrow \hat{X}_{t+1} \\ X_{t-m+2}, X_{t-m+3}, \ldots, X_{t+1} &\rightarrow \hat{X}_{t+2} \\ &\ldots \\ X_{t-m+n}, X_{t-m+n+1}, \ldots, X_{t+n-1} &\rightarrow \hat{X}_{t+n} \end{aligned}$$

Step 2: Compute the distance between the calculated ACE data sequence and the predicted one with

Manhattan Similarity.

$$d = \frac{1}{n}\sum_{i=1}^{n} |ACE_{t+i} - \hat{A}CE_{t+i}|$$

Step 3: Compare the distance with threshold θ. It is normal data if the distance is less than the threshold; i.e., $d < \theta$. Otherwise, it is regarded as attacked data.

These steps are also illustrated in Fig. 7.

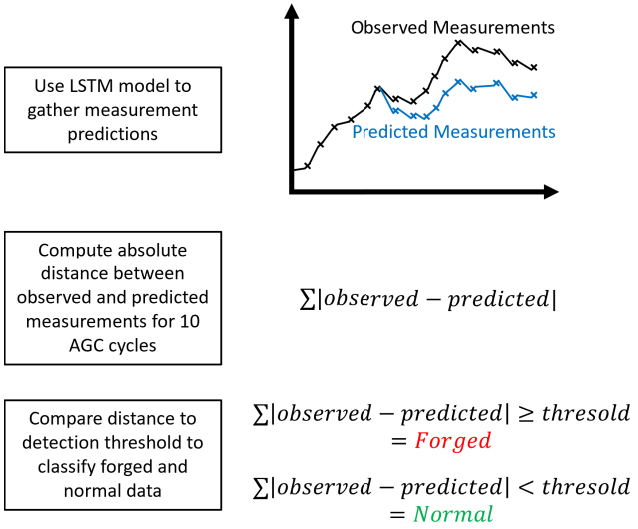


FIGURE 7. LSTM-based detection overview.

C. LSTM-BASED ATTACK LOCALIZATION

After detecting the existence of an attack, it is desired to know where the attack is from or which sensor is compromised. In the detection method, we detect attacks by finding abnormal ACE patterns. Following this idea, we can also detect which measurement is attacked by checking the measurements' patterns. Hence, to localize attacks, we not only predict ACE values but also predict the frequency and tie-line power flow values through the same LSTM model. When abnormal ACE data is detected, we will compare the predicted frequency and tie-line power flow values with the frequency and tie-line power flow measurements.

$$d_f = \frac{1}{n}\sum_{i=1}^{n} |f_{t+i} - \hat{f}_{t+i}|/f_{t+i}$$
$$d_P = \frac{1}{n}\sum_{i=1}^{n} |P_{t+i} - \hat{P}_{t+i}|/P_{t+i} \quad (8)$$

We can set thresholds for d_f and d_P. If d_f or d_P are larger than their respective threshold, that measurement is considered to be under attack.

D. AUTOMATIC THRESHOLD GENERATION FOR LSTM

The ideal detection threshold θ depends on many factors in the LSTM-based detection method. For example, we could use a relatively lower threshold for a larger dataset, and use a relatively higher threshold for a smaller dataset. This is because the LSTM model predictions typically become more accurate the larger the training data size, and as a result, we can use a lower threshold to detect more attacks without causing many false positives. Additionally, the larger the standard deviation in a dataset, the higher our detection threshold should be to reduce false positives. This is because there will likely be a larger gap between predicted and real data for a dataset with a larger standard deviation. While it is possible to find a proper threshold by manually trying different values and observing which one derives the best accuracy, in practice, this causes much overhead for system operators and, more importantly, system operators might not have the technical capability to do so.

To solve this problem, we developed a method to automatically determine a proper threshold based on the user's needs. The method takes the user's desired average false positive rate or average true positive rate as input. The user can specify one of the two requirements at a time. The method will split the dataset into three parts: training, pseudo-testing, and testing. The program will then train the LSTM model with the training dataset. With the trained model, it then generates predictions on the training dataset; i.e., it is predicting values on the same dataset that was used to train the LSTM model. The method will then calculate the absolute difference between the predicted and real data values. For every possible series of 8 AGC cycles (we found 8 works well in our experiments), i.e., 8 data values that are right next to each other, we calculate the sum of the absolute differences for each of those 8 cycle series. Now we have a large list of absolute differences for 8 cycle series. In this list, a majority of the list will likely be very small values. This is because the predictions on the training dataset will be very accurate. However, there will still be values that are relatively larger; this is because of LSTM's forget gate and general prediction faults. From this list of absolute differences, we will pick a specific percentile (we refer to this percentile as the *generation percentile*), and try using the corresponding absolute difference value as the detection threshold for the pseudo-testing dataset to see if it meets the user's required average false positive rate or true positive rate. If it meets the requirement, it is selected as the final detection threshold; if it does not meet the requirement, we will select a different generation percentile and repeat this process until the needed threshold is found. For example, if we pick 100 as the generation percentile, essentially the largest value from the list of absolute differences will be tried as the threshold. If we pick 0 as the generation percentile, essentially the smallest value from the list will be tried as the threshold. When trying the generation percentiles, we follow the descending order, i.e., from large to small percentiles. We follow this order because as the percentile decreases, the threshold value decreases. As the threshold decreases, more attacks can be detected (i.e., true positive rate increases), but more good data can be detected as attacks too (i.e., false positive rate increases). Following the descending order, we can

find the right threshold that meets the user's requirement on false positive rate while maintaining as high true positive rate as possible, or the right threshold that meets the user's requirement on true positive rate while maintaining as low false positive as possible. Our method automatically creates false data injection attacks in the pseudo-testing dataset to test the threshold values over the dataset.

The above method can also be used to automatically find the right threshold for attack localization based on user needs.

E. FOURIER TRANSFORM-BASED DETECTION AND LOCALIZATION

Besides the LSTM-based approach, we also propose another method, Fourier Transform-based data forgery detection and localization. Although it has slightly lower performance than the multi-feature LSTM model, it is simpler and has reasonably good performance as shown in evaluations.

Fourier Transform [29] can convert data from the time domain to the frequency domain to observe data patterns. The moving average is to calculate the average of consecutive data to smooth fluctuations and highlight patterns, which can make data sequence patterns more obvious. Thus, we can calculate the moving average of data and then convert its moving average to the frequency domain to observe patterns. If some unexpected changes occur, they can also be reflected in the moving average. The moving average of attacked ACEs and normal ones are shown in Fig. 8. The blue line shows normal data's moving average and the red one is the moving average with the scale attack ($\lambda_s = 0.2$) when the length of one attack period is 10 AGC cycles. It can be seen that the patterns of moving average are more obvious in the frequency domain, and the attacked data's moving average is more fluctuated than the normal ones.

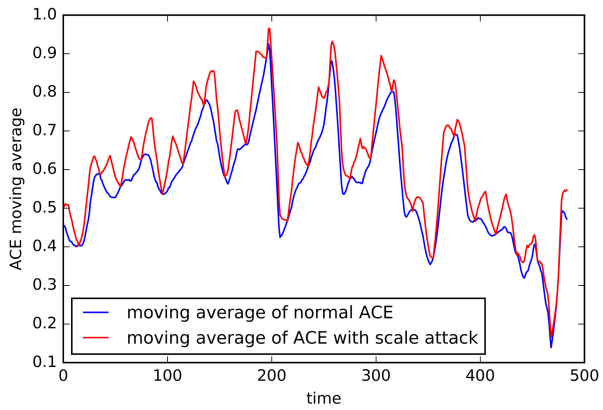


FIGURE 8. Moving average of normal ACE and ACE with scale attack.

In this detection method, we first calculate the moving average for each 10-data sequence, and then convert the moving average to the frequency domain and get the minimum transformed value (MTV) of each sequence. As shown in Fig. 9, the MTVs have significant differences. The MTV of a normal moving average is around 0.0 while the ones

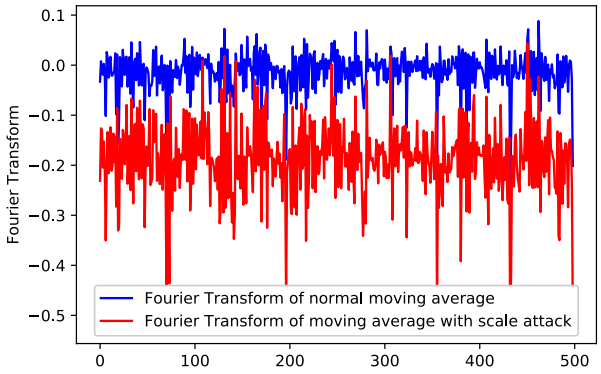


FIGURE 9. Fourier transform of moving average with scale attacks.

with scale attack is around -0.2. The attacked MTVs can be separated from normal ones by setting a threshold. If a data sequence's MTV is larger than the threshold, it is normal data. Otherwise, it is regarded as attacked data. Such a threshold can be set by observing the differences between MTVs of attacked data and normal data.

To localize attacks, we perform the same Fourier Transformation on tie-line power flow and frequency measurements. Then we can set a threshold for each to check whether the measurements' converted MTV values are under attack.

F. AUTOMATIC THRESHOLD GENERATION FOR FOURIER TRANSFORM

Similar to LSTM, we design a method to automatically generate a detection threshold for our Fourier transform detection method too. Users can specify the desired average true positive rate or false positive rate. The method will split the dataset into three parts: training, pseudo-testing, and testing. It will calculate the moving average for each 10-data sequence in the training dataset, and then convert the moving average to the frequency domain and get the minimum transformed value (MTV) of each sequence. All of these values are stored in a list. From this list, the method will pick different generation percentiles and try them over the pseudo-testing dataset until a threshold that meets the user needs is found. Different from the LSTM method, when trying the generation percentiles, the ascending order is followed. This is because the threshold values are usually negative in the Fourier transform-based method, and we detect an attack if the MTV is smaller than the threshold. As the percentile increases, the threshold value increases, and the true positive rate and false positive rate increase too.

V. DATASETS

We evaluate our solutions based on one synthetic dataset and two real datasets.

A. SIMULATED DATASET

A 5-bus power system [30] as shown in Fig. 10 was simulated, which is a typical power system with two interconnected control areas. The two control areas are connected by two

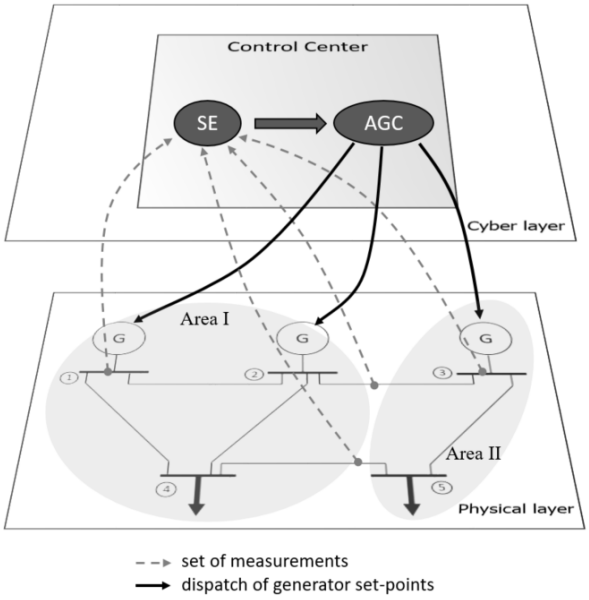


FIGURE 10. 5-bus system with 2 control areas.

tie-lines. Area I contains buses 1, 2 and 4, and Area II contains buses 3 and 5. Bus 4 and Bus 5 are the load buses for Control Area I and Area II respectively. Buses 1-3 are generator buses. Area I is equipped with two generators and Area II is equipped with one generator. The power system dynamics are modeled by using the structure-preserving load model [31], [32] and the well-known generator model with governor control [33].

Each control area is equipped with its own AGC, which sits in the control center. The control center periodically collects frequency and tie-line flow measurements, checks the measurements with a state estimation method, and then passes the measurements to AGC. AGC calculates ACE based on the measurement with Eq. (1) and then dispatches new set-points to generators.

The system was simulated with real load consumption data used at the load buses. Thus, the realistic ACE patterns were inserted into the simulated AGC system. Real time actual load measurements and the load forecast data from two areas in NY-ISO [34] were collected, and load deviation values for the system were generated by subtracting the forecast values from actual load values. The load deviations were scaled down to fit the parameters of the small simulated grid. The AGC system was simulated under these disturbances: deviations in loads, and generated realistic measurements of frequency, tie-line power flows, and ACEs. To obtain the attacked data, false data was injected into the tie-line power flow measurements based on the attack models in Section III-B and ACE was calculated by AGC with the attacked tie-line power flow data.

B. REAL DATASETS

To test our approaches, we collected two publicly available real datasets (denoted by *Real1* and *Real2*, respectively) from two electric utility organizations [26], [35]. The *Real1* dataset includes four years' ACE data, from the year 2012 to 2015, with about 2 million data records. Each record provides the ACE value and its date and time. The *Real2* dataset consists of the ACE data of 2016 and 2017, which has about 1 million records and the same data format as the *Real1* dataset. These datasets are considered as normal data without any attack. To generate the attacked data, we inject false data into ACE directly.

VI. EVALUATION

In this section, we evaluate the proposed methods. Additionally, we also implemented the method in [13] for comparison; we will refer to this method as the *baseline* method. It works by training a model to estimate how much false data was injected by a potential attack. We first test the detection and localization performance of the multi-feature LSTM-based method on the simulated dataset and compare it with the single-feature LSTM. Then we test the Fourier Transform-based method on the simulated dataset. Next, we compare the performances of the multi-feature LSTM, Fourier transform, and baseline methods on the simulated dataset. Lastly, we test the single-feature LSTM and Fourier Transform methods on the two real datasets. Since the real datasets have ACE data only, the multi-feature LSTM cannot be tested on it.

The neural network parameters and structures for multi-feature LSTM-based, single-feature LSTM-based, and baseline methods are detailed in Table 1.

TABLE 1. Neural network parameters and structures.

	Multi-Feature LSTM	Single-Feature LSTM	Baseline
Sequential Layers	1	1	1
LSTM Layers	100	6	6
Dense Layers	3	1	1
Output: Predicted Measurements	ACE, Tie-Line Power Flow, Frequency	ACE	False Data in ACE
Training Epochs	30	30	30
Training Batch Size	50	15	15

A. PERFORMANCE OF THE MULTI-FEATURE LSTM-BASED METHOD ON THE SIMULATED DATASET

The dataset used in the experiments is the simulated dataset as described in Section V-A. It includes about 1 million records and each record has ACE, frequency, tie-line power flow, and real load data. The dataset is chronologically split into

training dataset (67%) and testing dataset (33%). The LSTM model is built with a hidden layer with 100 neurons and an output layer and is trained with the training dataset. In the testing, the input data sequence length m is set as 5, and the predicted data sequence length n is set as 10 since the shortest attacked sequence which can negatively influence the system is 10 as discussed in [3]. For each attack, false data is injected into tie line power flow measurements for 10 cycles. To test the model's performance, we feed the attacked data and check the True Positive (TP) detection rate, which is defined as the fraction of attacks successfully detected. We also feed normal data without attacks into the model to see the False Positive (FP) detection rate, which is defined as the fraction of normal data sequences falsely detected as attacked data. We also test the localization rate, defined as the fraction of attacks localized correctly.

The setting of the threshold θ is critical. If the threshold is too low, some normal data sequences will be detected as attacked data. If the threshold is too high, the attacked data sequences may not be detected. The higher the threshold is, the lower the FP rate is. In the following, we set the threshold as 0.0375, which has an FP rate of less than 5%.

1) RANDOM ATTACK

In this experiment, we launched random attacks between λ_a and λ_b to tie-line power flow measurements periodically. Here we make $\lambda_a = -\lambda_b$.

The results are shown in Fig. 11. When λ is higher, the TP detection rate is also higher. Higher λ means the attacks have more significant modifications on ACE data and thus such attacks are easier to detect (note that these attacks also have a higher impact on the power grid). When $\lambda = 0.1$, the attack only change ACEs by less than 2.8% (see below), but more than 95% attacks can still be detected. The FP rates are under 1.3% and the localization rates are above 90%. Thus, the detection and localization performance is high.

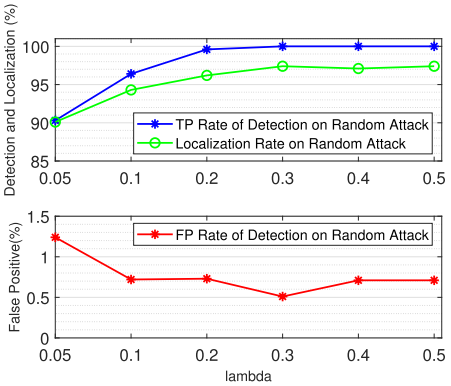


FIGURE 11. Detection and localization of multi-feature LSTM on random attack.

To further elaborate the detection, Fig. 14 shows the ACEs with random attacks when $\lambda_a = -0.1$ and $\lambda_b = 0.1$. The red line shows ACEs with random attacks and the blue line shows normal ACE data. The average of normal ACEs is about -3.5. The random attacks change ACEs by less than 0.1 on average, which is only about 2.8%. The LSTM model will calculate the distances between predicted ACEs and observed ACEs and then compare the distances with the threshold, as we discussed in Section IV-B. The calculated distances are shown in Fig. 17. The red bars show the distances between attacked ACEs and predicted ACEs, and the blue bars show the distances between normal ACEs and predicted ACEs. As aforementioned, the threshold is set as $\theta = 0.0375$. It can be seen that almost all the distances of normal ACEs are under the threshold while the distances of attacked ACEs are above the threshold, even if the attack only changes ACEs by 2.8%. This explains why the detection performance is high.

2) RAMP ATTACK

Fig. 12 shows the detection and localization results. When λ is higher, the detection is more accurate. When $\lambda >= 0.04$, more than 97% of attacks can be detected. The FP is under 3.5% and the localization is above 90%. To elaborate on the high performance, Fig. 15 shows ACEs with ramp attacks when $\lambda = 0.04$. On average, the ramp attacks with $\lambda = 0.04$ add about 0.2 to normal ACEs, and that changes ACEs by about 5%. Similar to random attacks, Fig. 18 shows that almost all the distance bars of attacked ACEs are above the threshold, while the distances of normal ACES are under the threshold.

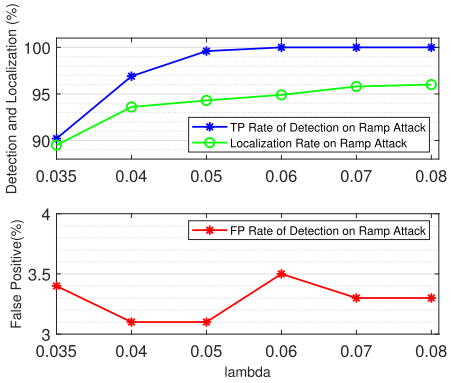


FIGURE 12. Detection and localization of multi-feature LSTM on ramp attack.

3) SCALE ATTACK

Fig. 13 shows the detection and localization results. When λ is higher, the detection rate increases. When $\lambda >= 0.04$, More than 98% of attacks can be detected. The FP is under 2% and the localization is above 94%. Fig. 16 shows the ACE with scale attacks when $\lambda = 0.04$. From the figure, it can be seen that the attacks only change ACEs by about 4%, but the threshold can still separate the distances of attacked ACEs from the distances of normal ACES as shown in Fig. 19.

4) MIN ATTACK

In this experiment, we launched Min attacks by replacing real ACEs with the minimum ACEs in the last 400 cycles.

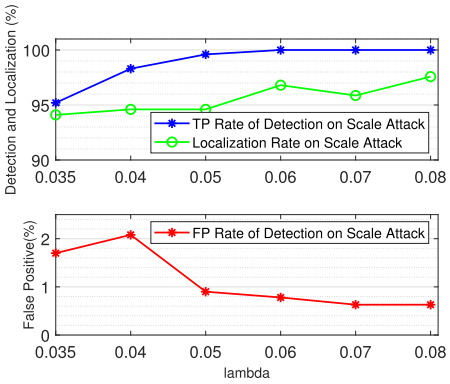


FIGURE 13. Detection and Localization of multi-feature LSTM on scale attack.

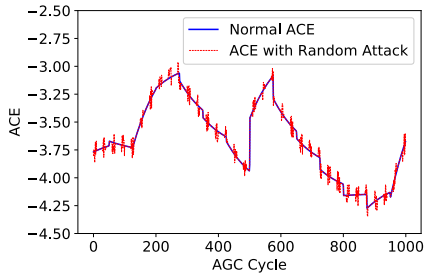


FIGURE 14. Normal ACE and ACE with random attack.

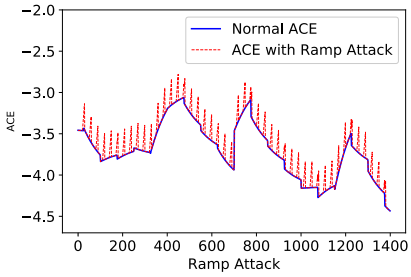


FIGURE 15. Normal ACE and ACE with ramp attack.

Fig. 21 shows the ACE with Min attacks. It can be seen that in some AGC cycles, the attacks are very obvious, while in other cycles the attacks can be negligible. This is because some ACE values are very close to the attacked value, and the Min attacks have almost no modification on real ACEs. In such situations, our method cannot detect the attacks, but there is also no need to detect them anyway since they do not have any impact on the power system. That is why the TP rate shown in Fig. 20 is not as high as other attacks, such as random attacks. The FP rate is still very low, which is about 0.8% and the localization rate is above 90%.

5) MAX ATTACK

In this experiment, we launched Max attacks by replacing real ACEs with the maximum ACEs in the last 400 cycles. Fig. 24 shows the ACE with Max attacks. It has similar trends and attributes with Min attacks. The TP rate is shown in Fig. 23,

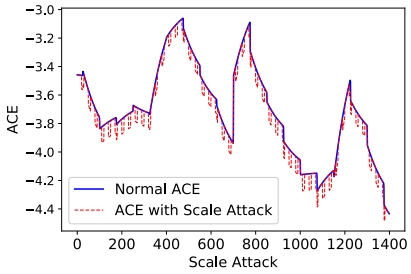


FIGURE 16. Normal ACE and ACE with scale attack.

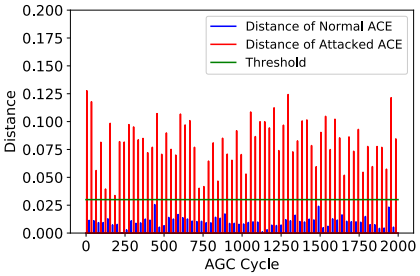


FIGURE 17. Distances of normal and random attacked ACE.

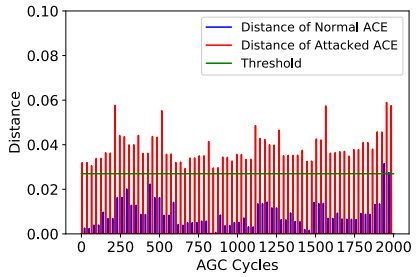


FIGURE 18. Distances of normal and ramp attacked ACE.

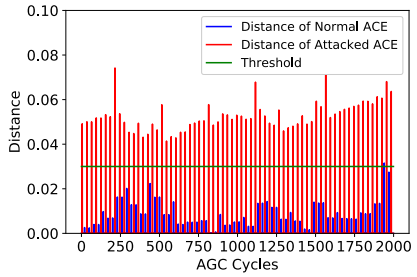


FIGURE 19. Distances of normal and scale attacked ACE.

which is similar to Min attacks. This is because in some AGC cycles, the real ACEs are close to the maximum attack data. Such attacks have no modification on real ACEs and thus have no impacts on power systems.

B. COMPARISON BETWEEN MULTI-FEATURE AND SINGLE-FEATURE LSTM ON THE SIMULATED DATASET

To understand whether considering multiple features achieves better performance, we compare the single-feature and the multi-feature LSTM-based detection methods on the

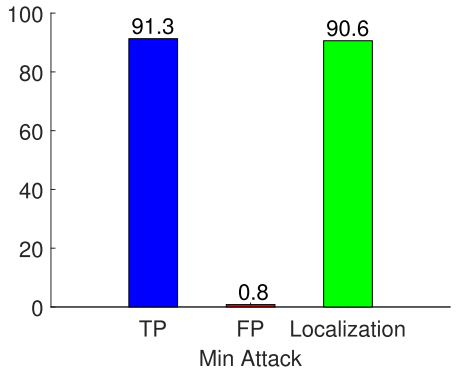


FIGURE 20. Multi-feature LSTM on min attack.

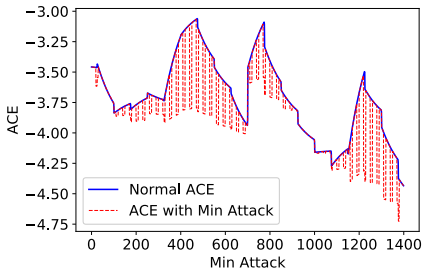


FIGURE 21. Normal ACE and ACE with min attack.

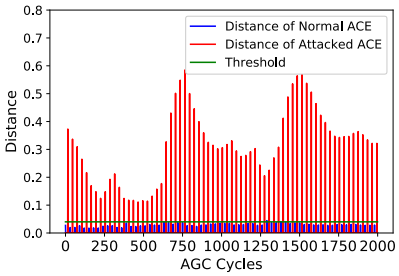


FIGURE 22. Distances of normal and min attacked ACE.

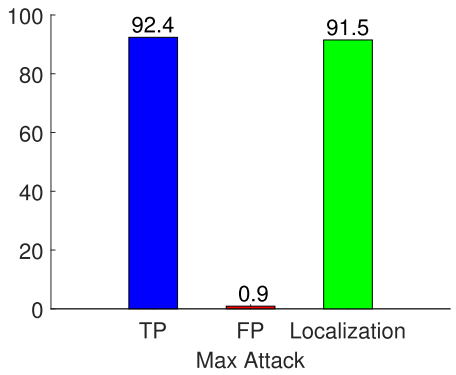


FIGURE 23. Multi-feature LSTM on max attack.

simulated dataset. For each attack, false data is injected into tie line power flow measurements for 10 cycles. For conciseness, we present the results for random attack, ramp attack, and scale attack. Similar trends hold for Min and Max attacks.

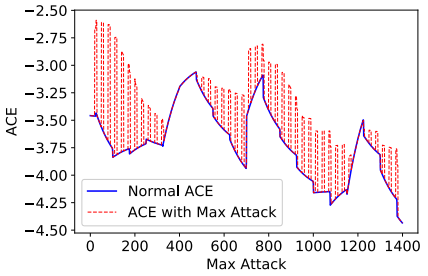


FIGURE 24. Normal ACE and ACE with max attack.

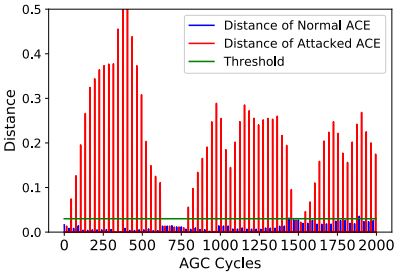


FIGURE 25. Distances of normal and max attacked ACE.

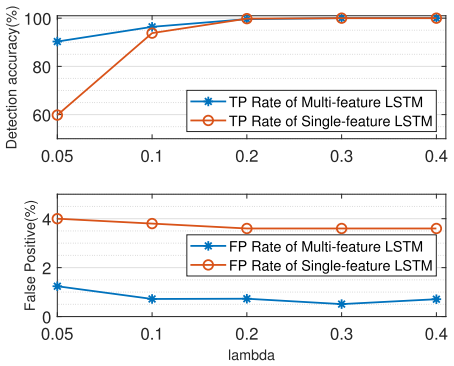


FIGURE 26. Multi-feature and single-feature LSTM on random attack.

1) COMPARISON ON RANDOM ATTACK

As shown in Fig. 26, the single-feature LSTM-based method also has good performance on random attack detection. When $\lambda = 0.1$, it can detect more than 95% of attacks. However, the multi-feature LSTM-based method outperforms it, especially when the attack parameter is low, i.e., when attacks are not obvious.

2) COMPARISON ON RAMP ATTACK

As it can be seen from Fig. 27, the single-feature LSTM-based method has great performance on ramp attack detection, but the multi-feature LSTM-based method still outperforms it in false positive rate.

3) COMPARISON ON SCALE ATTACK

As shown in Fig. 28, the single-feature LSTM-based method performs poorly on scale attack detection. When $\lambda = 0.05$, it can only detect about 50% of attacks. This is because scale attacks just scale the data value up or down, without

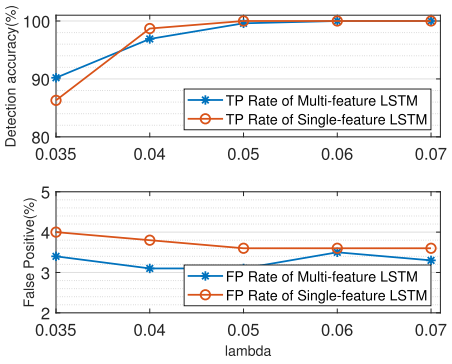


FIGURE 27. Multi-feature and single-feature LSTM on ramp attack.

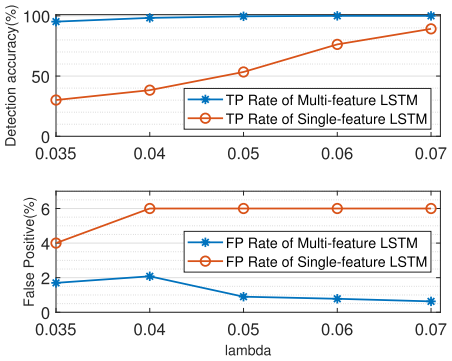


FIGURE 28. Multi-feature and single-feature LSTM on scale attack.

changing the ACE data sequence patterns much, and thus the single-feature LSTM model cannot differentiate them from normal data well. On the contrary, multi-feature LSTM considers other data features such as frequency and real load in addition to ACEs. Even though the ACE pattern is not changed much, attacks can still be detected with the help of other measurement data.

C. PERFORMANCE OF FOURIER TRANSFORM-BASED METHOD ON THE SIMULATED DATASET

In this section, we test how the Fourier transform-based method performs on attack detection and localization. We launched the same attacks as in the multi-feature LSTM testing. Fig. 29 shows the results under scale, ramp, and random attacks. The detection threshold is set as 0.00375 which has 4.3% FP rates. As shown in Fig. 29, the Fourier Transform-based method has high detection and localization accuracy for scale and ramp attacks, but it does not perform well in random attacks. It also performs well in Min and Max attacks, as shown in Fig. 30.

D. COMPARISON BETWEEN MULTI-FEATURE LSTM-BASED, FOURIER TRANSFORM-BASED, AND BASELINE METHODS ON SIMULATED DATASET

We compare the multi-feature LSTM-based, Fourier transform-based, and baseline methods on the simulated dataset. We present the results for random attack, ramp attack, and scale attack.

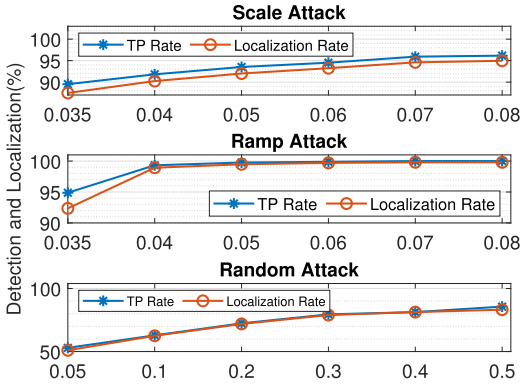


FIGURE 29. Detection and localization of Fourier transform.

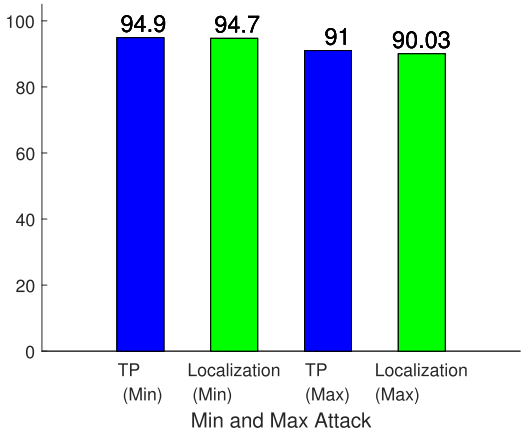


FIGURE 30. Fourier transform on min and max attack.

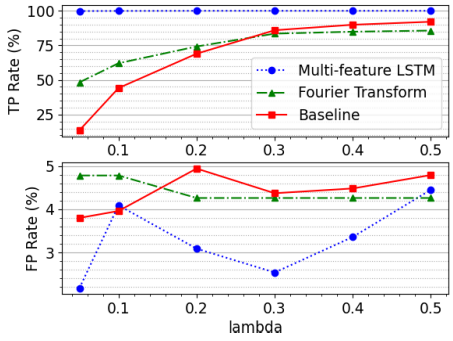


FIGURE 31. Comparison results on random attack.

1) COMPARISON ON RANDOM ATTACK

As shown in Fig. 31, the multi-feature LSTM-based method has a great performance on the random attack. The Fourier transform-based method has a poor performance when λ is small. Similarly, the baseline method also performs poorly when λ is small. The Fourier transform-based and baseline method's performance improves when λ is larger. Overall, the multi-feature LSTM-based method significantly outperforms the other two methods against the random attack.

2) COMPARISON ON RAMP ATTACK

As shown in Fig. 32, the multi-feature LSTM-based and Fourier transform-based methods have a great performance

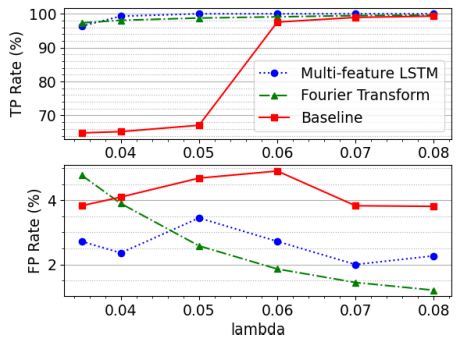


FIGURE 32. Comparison results on ramp attack.

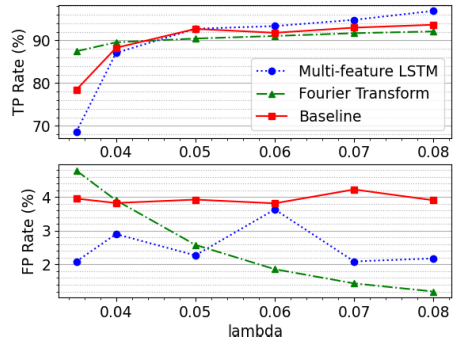


FIGURE 33. Comparison results on scale attack.

on the ramp attack. The baseline method has a reasonable performance when λ is small, and improves when λ is larger. Both of our proposed methods still outperform the baseline method against the ramp attack.

3) COMPARISON ON SCALE ATTACK

As shown in Fig. 33, against the scale attack, the multi-feature LSTM-based method outperforms the baseline method when λ is 0.05 or larger. When λ goes very small, both multi-feature LSTM's and the baseline method's performances significantly degrade probably because they are both based on machine learning and machine learning generally cannot handle extremely minor scaling attacks well. When λ is very small, the Fourier transform's performance also decreases a little, but it is better than the other two methods in true positive rate.

E. COMPARISON OF SINGLE-FEATURE LSTM AND FOURIER TRANSFORM-BASED METHODS ON REAL DATASETS

In this section, we test the LSTM-based method and Fourier Transform-based method on the real datasets. Since there is no frequency, tie-line power flow, and real load in the real dataset, we are not able to test the multi-feature LSTM-based method and hence focus on the single-feature LSTM. For the same reason, we are not able to test the localization performance and hence focus on detection. For each attack, false data is injected into ACE for 10 cycles. For the power system where the real dataset was generated, under random attack

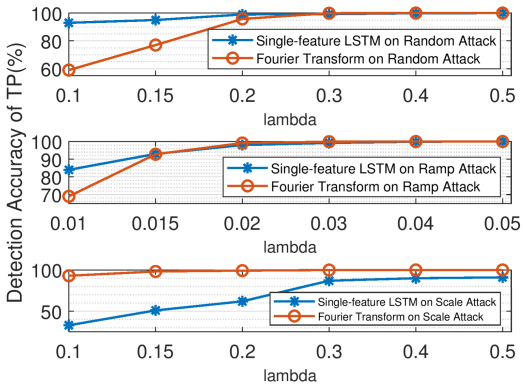


FIGURE 34. Single-feature LSTM and Fourier transform on the *Real1* dataset.

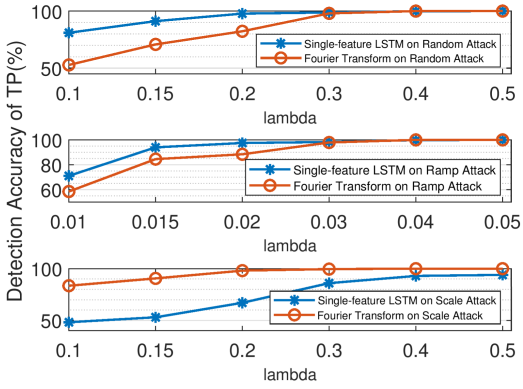


FIGURE 35. Single-feature LSTM and Fourier transform on the *Real2* Dataset.

with $\lambda_a = -0.1$, $\lambda_b = 0.1$, ramp attack with $\lambda_r = 0.1$, and scale attack with $\lambda_s = 0.1$, ACEs are changed by about 5%, 5%, and 10% on average, respectively. The detection results of the *Real1* dataset are shown in Fig. 34. The single-feature LSTM has better performance than Fourier Transform on random and ramp attacks. For example, when random attack's parameter $\lambda = 0.1$, single-feature LSTM can detect more than 90% of random attacks. However, it performs poorly on scale attacks, as we discussed in Section VI-B3. The Fourier Transform method is able to detect scale attacks effectively. Thus, we can use the single-feature LSTM-based method to detect random and ramp attacks and use Fourier Transform as the complementary method to detect scale attacks, when only the ACE data is available. Fig. 35 shows the detection results on the *Real2* dataset, which have similar performance and trends with the *Real1* dataset.

F. PERFORMANCE OF AUTOMATIC THRESHOLD GENERATION

In order to test automatic threshold generation for LSTM, we took the results of different generation percentiles' on random, scale, ramp, min, and max attacks for the multi-feature LSTM method, and averaged all collected true positive rates and false positive rates. We also averaged the attack localization accuracy. The results are shown in Fig. 36. As the

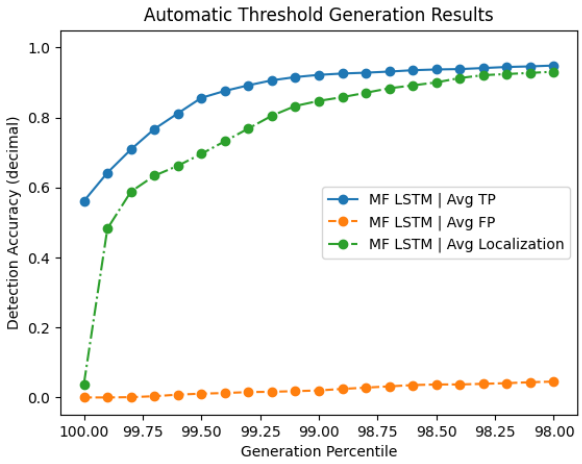


FIGURE 36. Automatic threshold generation results for multi-feature LSTM.

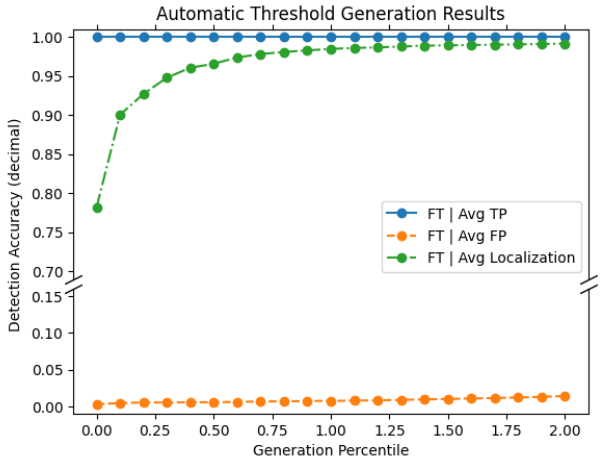


FIGURE 37. Automatic Threshold generation results for Fourier transform.

generation percentile decreases, the true positive rate and false positive rate both increase. Choosing a generation percentile of 98 achieves a good balance between the two.

In order to test automatic threshold generation for the Fourier transform-based method, we took the results of multiple generation percentiles' performance on random, scale, and ramp attacks. We then averaged all collected true and false positive rates. We also averaged the attack localization accuracy. The results are shown in Fig. 37. As the generation percentile increases, the true positive rate and false positive rate both increase. Choosing a generation percentile of 2.0 achieves a good balance between the two.

VII. CONCLUSION

We proposed Neural Network-based (multi-feature LSTM and single-feature LSTM) and Fourier Transform-based methods to detect and localize data forgery attacks in AGC. To make it easier for users to use the methods, we also designed an approach for automatically generating proper detection thresholds for the methods. We tested these methods against random, ramp, scale, min, and max attacks on real and simulated datasets. The experiments showed that both LSTM-based and Fourier transform-based methods have promising performance on attack detection and localization. Specifically, multi-feature LSTM has better performance than single-feature LSTM and the Fourier transform-based method. The single-feature LSTM-based method can detect most of the attacks but cannot detect scale attacks effectively, while the Fourier Transform-based method has good performance on scale attacks, which means they can be used as complementary detection methods in single-feature scenarios. Over the simulated dataset, for random attacks ($\lambda = 0.1$), ramp attacks ($\lambda = 0.04$), and scale attacks ($\lambda = 0.04$), the multi-feature LSTM-based method has an overall average of 94.9% true positive rate, 92.8% localization rate, and 1.6% false positive rate for the three attacks; the single-feature LSTM-based method has an overall average of 76.6% true positive rate and 4.6% false positive rate; the Fourier transform-based method has an overall average of 87.7% true positive rate, 87.1% localization rate, and 4.3% false positive rate.

ACKNOWLEDGMENT

The authors would like to thank Marija Ilic and Ana Jevtic for the synthetic dataset.

REFERENCES

[1] A. Gomez-Exposito and A. Abur, *Power System State Estimation: Theory and Implementation*. Boca Raton, FL, USA: CRC Press, 2004.

[2] L. Xie, Y. Mo, and B. Sinopoli, "Integrity data attacks in power market operations," *IEEE Trans. Smart Grid*, vol. 2, no. 4, pp. 659–666, Dec. 2011.

[3] R. Tan, H. H. Nguyen, Eddy. Y. S. Foo, X. Dong, D. K. Y. Yau, Z. Kalbarczyk, R. K. Iyer, and H. B. Gooi, "Optimal false data injection attack against automatic generation control in power grids," in *Proc. ACM/IEEE 7th Int. Conf. Cyber-Phys. Syst. (ICCPS)*, Apr. 2016, pp. 1–10.

[4] (2021). *Sophisticated Hackers Could Crash the U.S. Power Grid, But Money, Not Sabotage, is Their Focus*. [Online]. Available: https://www.utilitydive.com/news/sophisticated-hackers-could-crash-the-us-power-grid-but-money-not-sabotag/603764/

[5] (2022). *Cyber Attacks on the Power Grid*. [Online]. Available: https://www.ironnet.com/blog/cyber-attacks-on-the-power-grid

[6] (2023). *3 Alarming Threats to the U.S. Energy Grid—Cyber, Physical, and Existential Events3 Alarming Threats to the U.S. Energy Grid—Cyber, Physical, and Existential Events*. [Online]. Available: https://www.forbes.com/sites/chuckbrooks/2023/02/15/3-alarming-threats-to-the-us-energy-grid–cyber-physical-and-existential-events/?sh=4d573cb7101a

[7] K. McClanahan, J. Fan, Q. Li, and G. Cao, "Protecting control commands using low-cost em sensors in the smart grid," in *Proc. IEEE Conf. Commun. Netw. Secur. (CNS)*, Oct. 2023.

[8] M. L. Uwibambe, Y. Pan, and Q. Li, "Fuzzing for power grid systems: A comparative study of existing frameworks and a new method for silent crash detection in control devices," in *Proc. IEEE Design Methodologies Conf. (DMC)*, Sep. 2023.

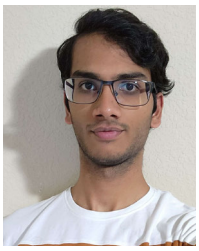
[9] S. Sridhar and M. Govindarasu, "Model-based attack detection and mitigation for automatic generation control," *IEEE Trans. Smart Grid*, vol. 5, no. 2, pp. 580–591, Mar. 2014.

[10] M. Q. Ali, R. Yousefian, E. Al-Shaer, S. Kamalasadan, and Q. Zhu, "Two-tier data-driven intrusion detection for automatic generation control in smart grid," in *Proc. IEEE Commun. Netw. Secur. (CNS)*, Oct. 2014, pp. 292–300.

[11] F. Akbarian, A. Ramezani, M. Hamidi-Beheshti, and V. Haghighat, "Advanced algorithm to detect stealthy cyber attacks on automatic generation control in smart grid," *IET Cyber-Phys. Syst., Theory Appl.*, vol. 5, no. 4, pp. 351–358, Dec. 2020.

[12] A. S. L. V. Tummala and R. K. Inapakurthi, "A two-stage Kalman filter for cyber-attack detection in automatic generation control system," *J. Mod. Power Syst. Clean Energy*, vol. 10, no. 1, pp. 50–59, Jan. 2022.
[13] C. Chen, Y. Chen, J. Zhao, K. Zhang, M. Ni, and B. Ren, "Data-driven resilient automatic generation control against false data injection attacks," *IEEE Trans. Ind. Informat.*, vol. 17, no. 12, pp. 8092–8101, Dec. 2021.
[14] A. Ayad, M. Khalaf, M. Salama, and E. F. El-Saadany, "Mitigation of false data injection attacks on automatic generation control considering nonlinearities," *Electr. Power Syst. Res.*, vol. 209, Aug. 2022, Art. no. 107958.
[15] A. S. Musleh, G. Chen, Z. Y. Dong, C. Wang, and S. Chen, "Attack detection in automatic generation control systems using LSTM-based stacked autoencoders," *IEEE Trans. Ind. Informat.*, vol. 19, no. 1, pp. 153–165, Jan. 2023.
[16] S. D. Roy, S. Debbarma, and A. Iqbal, "A decentralized intrusion detection system for security of generation control," *IEEE Internet Things J.*, vol. 9, no. 19, pp. 18924–18933, Oct. 2022.
[17] Z. Chen, J. Zhu, S. Li, and T. Luo, "Detection of false data injection attack in automatic generation control system with wind energy based on fuzzy support vector machine," in *Proc. 46th Annu. Conf. IEEE Ind. Electron. Soc.*, Oct. 2020, pp. 3523–3528.
[18] Z. Chen, J. Zhu, H. Dong, W. Wu, H. Zhu, and C. He, "Detection of false data injection attack in automatic generation control system with electric vehicles based on fuzzy long short-term memory," in *Proc. IEEE Power Energy Soc. Gen. Meeting (PESGM)*, Jul. 2021, pp. 1–5.
[19] Y. W. Law, T. Alpcan, and M. Palaniswami, "Security games for risk minimization in automatic generation control," *IEEE Trans. Power Syst.*, vol. 30, no. 1, pp. 223–232, Jan. 2015.
[20] A. Jevtic, F. Zhang, Q. Li, and M. Ilic, "Physics-and learning-based detection and localization of false data injections in automatic generation control," in *Proc. IFAC Symp. Control Power Energy Syst. (CPES)*, 2018, vol. 51, no. 28, pp. 702–707.
[21] Y. He, G. J. Mendis, and J. Wei, "Real-time detection of false data injection attacks in smart grid: A deep learning-based intelligent mechanism," *IEEE Trans. Smart Grid*, vol. 8, no. 5, pp. 2505–2516, Sep. 2017.
[22] F. Zhang and Q. Li, "Deep learning-based data forgery detection in automatic generation control," in *Proc. IEEE Conf. Commun. Netw. Secur. (CNS)*, Oct. 2017, pp. 400–404.
[23] F. Zhang, "Countering cybersecurity vulnerabilities in the power system," Ph.D. dissertation, Dept. Comput. Sci. Comput. Eng., Univ. Arkansas, Fayetteville, AR, USA, 2019. [Online]. Available: https://scholarworks.uark.edu/etd/3556
[24] N. Jaleeli, L. S. VanSlyck, D. N. Ewart, L. H. Fink, and A. G. Hoffmann, "Understanding automatic generation control," *IEEE Trans. Power Syst.*, vol. 7, no. 3, pp. 1106–1122, Aug. 1992.
[25] Y.-L. Huang, A. A. Cárdenas, S. Amin, Z.-S. Lin, H.-Y. Tsai, and S. Sastry, "Understanding the physical and economic consequences of attacks on control systems," *Int. J. Crit. Infrastruct. Protection*, vol. 2, no. 3, pp. 73–83, Oct. 2009.
[26] (2017). *PJM ACE Data*. [Online]. Available: http://www.pjm.com/markets-and-operations/etools/oasis/system-information/historical-area-control-error-data.aspx
[27] D. P. Mandic and J. Chambers, *Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures and Stability*. Hoboken, NJ, USA: Wiley, 2001.
[28] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.
[29] E. O. Brigham and R. E. Morrow, "The fast Fourier transform," *IEEE Spectr.*, vol. S-4, no. 12, pp. 63–70, Dec. 1967.
[30] X. Z. Liu, "Structural modeling and hierarchical control of large-scale electric power systems," Ph.D. dissertation, Massachusetts Inst. Technol., Cambridge, MA, USA, 1994.
[31] M. D. Ilic, L. Xie, U. A. Khan, and J. M. F. Moura, "Modeling of future cyber–physical energy systems for distributed sensing and control," *IEEE Trans. Syst., Man, Cybern., A, Syst. Hum.*, vol. 40, no. 4, pp. 825–838, Jul. 2010.
[32] A. R. Bergen and D. J. Hill, "A structure preserving model for power system stability analysis," *IEEE Trans. Power App. Syst.*, vol. PAS-100, no. 1, pp. 25–35, Jan. 1981.
[33] M. D. Ilic and J. Zaborszky, *Dynamics and Control of Large Electric Power Systems*. New York, NY, USA: Wiley, 2000.
[34] (2018). *Real-Time Actual Load Data Reports*. [Online]. Available: http://www.nyiso.com/public/markets operations
[35] (2017). *SPP ACE Data*. [Online]. Available: https://marketplace.spp.org/pages/ace-chart

FENGLI ZHANG received the B.E. degree from the University of Science and Technology of China, and the Ph.D. degree from the University of Arkansas, Fayetteville. Her research interest includes cybersecurity in the smart grid.



YATISH DUBASI received the B.S. degree from the University of Arkansas, Fayetteville, where he is currently pursuing the Ph.D. degree in computer science. His research interest includes privacy-preserving machine learning.

WEI BAO received the B.S. degree in industrial engineering from the Huazhong University of Science and Technology, and the M.S. and Ph.D. degrees from the University of Arkansas. His research interests include privacy-preserving machine learning and the secure outsourcing of computations.

QINGHUA LI (Member, IEEE) received the B.E. degree from Xi'an Jiaotong University, the M.S. degree from Tsinghua University, and the Ph.D. degree from The Pennsylvania State University. In 2013, he joined the University of Arkansas, where he is currently an Associate Professor and the 21st Century Research Leadership Chair of the Department of Electrical Engineering and Computer Science. His research interests include cybersecurity and privacy, smart grids, mobile computing, cloud computing, and artificial intelligence. He received the NSF CAREER Award, in 2018.