Application of Long Short-Term Memory (LSTM) Neural Network for the estimation of communication network delay in smart grid applications

Ronak Feizimirkhani, van Hoa Nguyen, Yvon Bésanger, Quoc Tuan Tran, Antoneta Iuliana Bratcu, Antoine Labonne, Thierry Braconnier

Application of Long Short-Term Memory (LSTM) Neural Network for the estimation of communication network delay in smart grid applications

Ronak Feizimirkhani[1], Van Hoa Nguyen[1], Yvon Bésanger[1], Quoc Tuan Tran[2], Antoneta Iuliana Bratcu[3], Antoine Labonne[1], Thierry Braconnier[1]

[1] Univ. Grenoble Alpes, CNRS, G*renoble INP (*Institute of Engineering Univ. Grenoble Alpes*), G2Elab,* Grenoble, France
[2] Univ. Grenoble Alpes, INES, CEA, LITEN, Le Bourget du Lac, France
[3] Univ. Grenoble Alpes, CNRS, G*renoble INP (*Institute of Engineering Univ. Grenoble Alpes*), GIPSA-lab,* Grenoble, France

***Abstract*— Vast integration of new technologies to enable smart control of the power grid requires a reliable, efficient and resilient communication infrastructure. Today, many communication protocols (*e.g.*, IEC 61850, OPC UA, Modbus, Internet, WiMAX, 4G, Wi-Fi, *etc.*) and technologies (*e.g.*, PLC, GSM, Optic fiber, RF radio mesh, Cellular, *etc.*) are established for the smart grid applications. In case of the stability guarantee of smart grid, the Quality of Service (QoS) is a challenge to be considered. One of the major concerns in data delivery over the network is a low latency message transmission to ensure the time critical tasks, *e.g.*, control and protection tasks. In this context, the main contribution of this paper is to propose a model methodology for the communication network delay in smart grid applications. To be compatible with the further goal of delay predictive compensation method, the present paper proposes a message transmission delay estimation method using Long Short-Term Memory (LSTM) neural network. To this end, Python is the chosen programming language, including its required libraries for the considered application. Delay values measured on a real-time HV/MV substation application are used as input data for validation purpose.**

Keywords— smart grid, Internet, message transmission delay, time-series forecasting, Long Short-Term Memory (LSTM)

I. INTRODUCTION

Current power grid is moving towards a modern grid, which benefits from innovative products and services such as: Distributed/Renewable Energy Resources, energy storage technologies, and intelligent control-protection system. Modern power system properly named "Smart Grid" benefits of the Information and Communication Technologies (ICTs) to deliver real-time information and near-instantaneously balance the demand-response on the electrical grid. Thus, smart grids provide: 1) self-healing technologies, 2) distributed control and protection, 3) self-monitoring, 4) two-way communication.

Control, supervision and protection of the smart grid are dependent on the quality of the assigned ICT infrastructure (*i.e.*, Quality of Service – QoS). So, any information abruption (*e.g.*, data loss, data transmission delay, security attack, failure, *etc.*) can affect the functionality and stability of the power system. To this end, numerous researches are done to study different aspects of the QoS. Here, focus is on the real-time performance of the communication system, which transmits information between power devices (*e.g.*, critical measured values, protection commands, fault detection, device status, *etc.*).

Different works Study the impacts of message transmission delay on the accurate functionality of the power system, even instability, major faults and equipment damages. [1] proposes a testing framework for integrating Supervision, Control and Data Acquisition (SCADA) system with an advanced Real-time Simulation (RTS) and Power-Hardware-In-Loop (PHIL) in a cross-infrastructure manner. The impacts of the communication latency are also analyzed on an advanced voltage and frequency restoration in an isolated Micro Grid (MG). [2] considers a scenario where Distributed Generators (DGs) communicate through LANs and have access to the Internet to enable the remote control of MGs *via* cloud servers. The control objectives are frequency/voltage restoration and proportional power sharing. Remote controls over the islanded MGs are delayed, and it is shown that the MG frequency and voltage start to oscillate, so, the MG system in on the edge of instability.

According to the various types of communication technologies and protocols, there are different estimation methods proposed through the literature. The evaluation of communication latency can be categorized by the underlying protocols of Local Area Network (LAN) and Wide Area Network (WAN) while applying a compatible method such as Network Calculus Theorem, Markov chain, queuing theory, machine learning, even simulation platforms. [3] proposes a combination of network calculus and measurements. To construct a real service curve model, and produce accurate delay bounds, it is validated against the measured values over a real Ethernet network based on IEC 61850. In this approach, the latency component of service curve is extracted based on measuring values: by injecting sampled value messages at low rate and measuring the maximum delay value which is the non-queuing delay (*i.e.*, physical delay property of the related switch). A new mathematical model and methodology are proposed by [4] to evaluate the performance of large-scale RF-mesh networks. An analytic formulation for delay is derived based on Markov-modulated modeling of the system. The proposed methodology is applied to a large-scale network of several thousands of nodes, and numerical results report that a wide variety of performance evaluations are enabled. Various traffic scenarios were considered in the analysis in order to evaluate the feasibility of a wider range of possible applications.

Various works try to precisely analyze the real-time performance of communication networks over WAN protocols. The communication delay with Wide-Area

978-1-6654-3613-7/21/$31.00 ©2021 IEEE

Measurement Systems (WAMS) for multi-energy complementary systems is calculated in [5]. First, a three-layer hierarchical distributed topology structure is constructed for WAMS communication network (*i.e.*, process modeling, node modeling, network modeling). Then, using the dynamic characteristics of time-division grading and sampling intervals, the network calculus algorithm was applied to assess the latency of the dynamic PMU data flow to the monitoring center. Finally, an OPNET-based three-layer communication network simulation model was established to show the effectiveness of the proposed model.

There are numerous works on delay estimation over Internet: [6] focuses on the stochastic delay analysis to satisfy the delay requirement of the space traffic and guide the network configuration in satellite data relay networks (SDRNs) communicating over 6G communication network. To accurately model the data offloading process in SDRNs, authors build a series-parallel queuing model with through and cross traffic while considering the propagation delay. Next, a propagation delay embedded min-plus convolution method is proposed using stochastic network calculus and a Markov-chain method based on Monte Carlo simulation. The method effectiveness is verified through many simulations in STK and MATLAB®, and some real use cases. Generally, [7] investigates the communication and networking challenges of fog computing in smart grids, and comprehensively discusses a 5G-based distributed network scheme. Using this architecture, communication delay can be reduced between different smart devices (SDs) deployed across the geographical location within the Home Area Network (HAN), Neighborhood Area Network (NAN), or WAN. [8] proposes a composable analysis approach called Local Arrival Curve (LAC) method based on the network calculus. This method first partitions the system into smaller ones according to the multiplexing nodes' positions, calculates the local delay bound based on the aggregate arrival curve and service curve, then sums up the local results to finally get the end-to-end delay bound.

Communication delay is also estimated using forecasting methods by time-series and machine-learning methods. In [9], a deep neural network (DNN) is adopted for predicting delay in IEEE-802.15.4-standard-based applications in Internet of Things (IoT). Results reveal that DNN achieves a prediction accuracy of over 98% in predicting delay.

Despite a vast research on the subject of delay estimation in smart grid, there is a lack of an accurate framework to model and analyze the message transmission delay while message is sent and forwarded through a control/protection loop (*i.e.*, from a controller unit to an actuator). Through this work with the goal of delay estimation/compensation, the communication channel is assumed to be unknown, meaning that there is no information of the channel properties and functionality of the protocols. Therefore, it is decided to base our work on machine-learning methods. It is possible to further analyze delay behavior online and to do the compensation according to provided delay values. Hence, a proper method could be a forecasting time-series method to describe the delay behavior whereas it may be variable. In addition, delay and its impacts on the under-study system (*i.e.*, smart grid communication network) can be compensated using a predictive control method. In this context, this paper focuses on the pair of predictive estimation-compensation.

The rest of this paper is continued in Section II by an introduction of the proposed delay estimation methodology and the communication channel architecture. The choice of LSTM model for delay analysis and its structure are detailed in Section III. The under-study platform is presented in Section IV, as well as the procedure of LSTM implementation using Python open-source software libraries. Finally, this paper is concluded by Section V.

II. DELAY ESTIMATION METHODOLOGY

As part of the enormous changes through smart grid, this reserach aims at bringing legacy communication systems to the *cloud* computing. So, there is a need to evaluate the message transmission delay and to compensate its impacts on the functionality of the applications that are deployed on the cloud server (*i.e.*, secondary and coordinated control, state estimation and optimization, Artificial Intelligence (AI) for forecasting, *etc.*). Predictive methods are the choice of the compensation phase and require a compatible forecasting method for delay estimation phase. To do so, delay values are sampled over time sequentially over time to be modeled using time-series forecasting methods.

A. Time series

A time series represents the observations on a variable which are taken sequentially in time. Data are measured at specific time t as a set of n observations, which is a sample realization from an infinite population of such samples: $y_1, y_2, \dots, y_n$. The inherent attribute of a time series is that, typically, adjacent observations are dependent. The nature of this dependency is an issue of practical interest. Time-series analysis takes advantage of techniques to analyze this dependency which requires to develop stochastic and dynamic models.

Analyzing methods help to build, identify, fit, and check models for time series and dynamic systems. These methods detect and explore the linear relationship existing through the current values, historical data and exogenous factors. Time-series analysis methods can be used for forecasting of future values of a time series from current and past values (*i.e.*, General Additive Models (GAMs) [10]). This research considers discrete-time (sampled-data) systems where delay observations occur at equally-spaced intervals of time. Time-series analysis is typically used in statistics, signal processing, pattern recognition, econometrics, mathematical finance, weather forecasting, control engineering, communication engineering, *etc*.

To model the behavior of transmission delay, we consider the sampled delay measurements of a conversation between two end points as the under-study time-series data.

B. Architecture of the communication channel

A message sent by its source is received and forwarded several times by devices located in the control/protection loop to reach its destination. As mentioned before, through this work transmission is provided by different communication protocols/technologies. Any time, a message passes through a device in the middle of its trajectory, it is decoded and recoded in a new communication protocol to continue. So, the communication channel is assumed to be a series of sub-channels as illustrated in Figure 1.

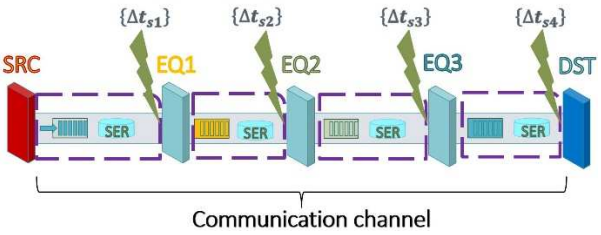

Figure 1 Considered communication channel with different devices through the path (SRC: Source, EQ: Equipment, DST: Destination, SER: Service).

Therefore, the first step is to estimate the transmission delay for each sub-channel, and finally the total transmission delay is the sum of all individual delays. In this case, each subsection includes a sender (or repeater) and a receiver, and delay values are sampled at the receiving point of each sub-channel. By applying the estimation process, delay is estimated and predicted.

III. MODEL DEVELOPEMENT

In this work, we are interested to apply descriptive methods over the observations (sampled delay values) to model the behavior of current value according to the historical data and some exogenous factors. Moreover, we aim at applying forecasting methods on the observed time series, so it is possible to use predictive delay compensators.

A. Choice of the forecasting method

Among different possible time-series forecasting methods, Auto Regressive Integrated Moving Average (ARIMA) model and Long Short-Term Memory (LSTM) model appear to be as two reasonable choices. According to the literature review, a comparison is done to select the most suitable method for the estimation of message transmission delay that fits particularities of Internet communication.

The state-of-the-art applications of the mentioned forecasting methods are compared according to their performance in three main fields: smart grid applications (data traffic, load consumption), communication networks (network traffic, communication delay), and economics (econometric data sets). This comparison is summarized in Table 1.

Based on a comparison through the literature review, LSTM is more flexible and adaptable in case of the Internet delay estimation, because of its capacity to learn long-term dependencies and remembering information for a long period of time – as the Internet delay is variable, this could improve the model accuracy.

B. Long Short-Term Memory (LSTM) neural network

LSTM is an evolution of Recurrent Neural Network (RNN) architecture in the field of deep learning. LSTM was introduced by Hochreiter and Schmidhuber [11] in order to enhance RNN drawbacks using some special units in addition to the standard units. LSTM units include "memory cells" that can maintain information for a long period.

An RNN consists of an input layer, one or more hidden layers, and an output layer. RNN is a chain of repeating modules as a memory to save important information from previous processing steps. RNN includes a feedback loop that allows accepting a sequence of inputs. It means the output of the previous step (t-1) is fed back into the network to influence the current output (t) and all the subsequent steps (t+1, *etc.*).

Table 1 A summary of the comparison among the characteristics of ARIMA and LSTM.

ARIMA	LSTM
Stationarity needed	
No parameter set up	Weights and biases setting
Linear relation only	Model nonlinearity of data
Simple implementation	
Fast run time	
	A large sample of data needed
Easier multivariable data management	
Self-similarity by FARIMA	Self-similarity is considered through data
Less accurate if fluctuated Internet delay	Less accurate if fluctuated Internet delay
	Auto-training and auto-adaptability
	Less Minimum Squared Error
	More accurate for Internet application (time-variable)
	Less accurate for a bigger prediction horizon

The sequential architecture of a typical RNN is shown in Figure 2. To solve some limitations of RNN – vanishing gradient problem, difficult training, short-memory – LSTM is a solution.

LSTM is chain-structured as all the RNNs but modified to learn long-range dependencies. Unlike the repeating module in a standard RNN, with a single layer, LSTM repeating module has four interacting layers. Each neuron of RNN structure is replaced by a memory unit (*i.e.*, cells) in LSTM structure. A unit contains a neuron with a recurrent self-connection. The activations of those neurons within the memory units are the state of the LSTM network.

At each time step, the memory unit receives input from the other memory units. It then computes how much of the input to forward to the neuron, how much of the neuron's previous activation to keep, and how much of its activation to output. LSTM contains states and gates in a cell. LSTM internal gating mechanism controls the memorizing process by regulating the flow of information. Gates can store, write and read information. There are three types of gates in the LSTM structure: input gate, forget gate, and output gate.

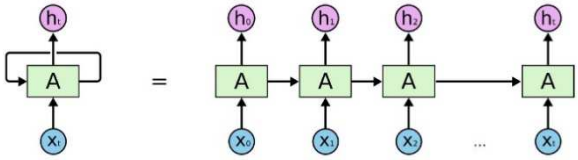

Figure 2 Sequential processing in an RNN [12].

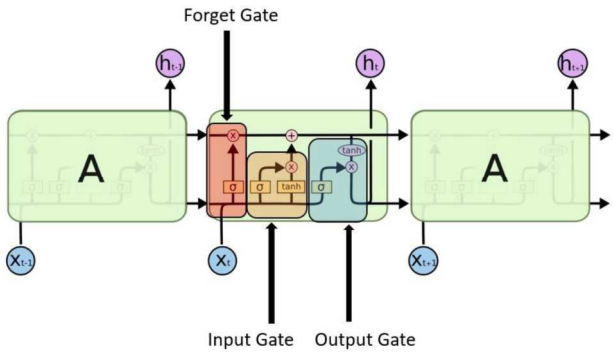

Figure 3 The architecture of LSTM neural network [13].

As illustrated in Figure 3, we compute the output of a memory unit by taking its state, applying an activation function, and multiplying the result by the output gate that says how much activation to output. The vector of outputs from all memory units is the output of the LSTM network.

Two states output to the next cell: cell state and hidden state. Cell state is the main data flow that allows information to flow forward while it remains intact.

A forget gate is described by a vector function f_t with values in the range of 0 to 1, corresponding to each number in the cell state, C_{t-1}, as follows:

$$f_t = \sigma(W_f[h_{t-1}, X_t] + b_f) \quad (1)$$

where σ is the sigmoid function, W_f and b_f are the weight matrix and bias of the forget gate, respectively. Function f_t decides which information to keep or remove. Information from the previous hidden state (h_{t-1}) and the current input (X_t) are passed through a sigmoid function (σ), whose values are between 0 and 1. The closer to 0 means to forget, and closer to 1 means to keep.

An input gate decides which information of the new input X_t is important to store in the cell state and to update the cell state. Two activation functions are involved in this step: sigmoid function decides whether new information should be stored or ignored (0 or 1), and a *tanh* function that assigns weights to the values passed through, deciding their level of importance (-1 to 1). The multiplication of these values updates the new cell state. So, this new memory is added to the old one, C_{t-1}, and result in C_t.

$$i_t = \sigma(W_i[h_{t-1}, X_t] + b_i), \quad (2)$$

$$N_t = tanh(W_n[h_{t-1}, X_t] + b_n), \quad (3)$$

$$C_t = C_{t-1}f_t + N_t i_t, \quad (4)$$

where C_{t-1} and C_t are the cell states at time $t-1$ and t, respectively, while W_n and b_n are the weight matrix and bias of the cell state, respectively.

An output gate provides output h_t that is filtered based on the output cell state O_t. Sigmoid function decides for the important information to make the output. Then, the sigmoid output O_t is multiplied by the new values of *tanh* function provided by the cell state C_t in the range of -1 to 1.

$$O_t = \sigma(W_0[h_{t-1}, X_t] + b_0), \quad (5)$$

$$h_t = O_t \tanh(C_t), \quad (6)$$

where W_0 and b_0 are weight matrices and bias of the output gate, respectively.

C. Model evaluation criteria

To evaluate the performance of the LSTM models, Mean Squared Error (MSE) is an often used statistical method to compare the predicted values against the observed ones. MSE is used to rate how accurately the predicted values fit over the observed ones.

$$MSE = {}^1/_n \sum_{t=1}^{n} (Y_t - \hat{Y}_t)^2, \quad (7)$$

where Y_t represents the observed value and $\hat{Y}_t$ is the predicted value. The squaring means that larger mistakes result in more error than smaller mistakes, meaning that the model is penalized for making larger mistakes.

IV. Model Design and Application

Open-source libraries are properly related to the case study considered in this research. Python [14] is selected as the programming language. In addition, NumPy [15], Pandas [16], Matplotlib [17] libraries are imported for processing, management and visualization data.

Here, LSTM model illustrated in Figure 3 is implemented using TensorFlow [18], which is an open-source software library provided by Google for its internal use. It is a symbolic math library constructed for machine learning, deep learning, and numerical computation based on dataflow graphs and differentiable programming. However, this framework is sufficiently comprehensive to be applicable to a wide variety of domains.

A. Model application procedure

In order to apply the LSTM model over the measured transmission delay, the following steps are performed sequentially:

a) The dataset is normalized. Normalization is a rescaling of the data from the original range so that all values are within the range between 0 and 1.

b) Split data into training dataset with 70% of the observations and test dataset with the 30% of observations.

c) Convert the array of values into the matrix form.

d) Creat and fit the LSTM network over the training dataset.

e) Make predictions over the training dataset, and over the test dataset.

f) Invert the predictions before calculating error scores to ensure that performance is reported in the same units as the original data.

g) Calculate the MSE as a loss function to evaluate the model accuracy.

h) Plot the error monitoring, and the prediction values against the measured values.

B. Validation test bench: communication over cloud

This work aims to transfer the legacy systems to the Cloud computing in the framework of smart grid development. Delay of message transmission over cloud can affect the applications that communicate over a cloud server (i.e., secondary and coordinated control, state estimation and optimization, AI for forecasting, etc.). Hence, it is needed to evaluate the communication latency in order to compensate data transfer delay and consequently its impacts.

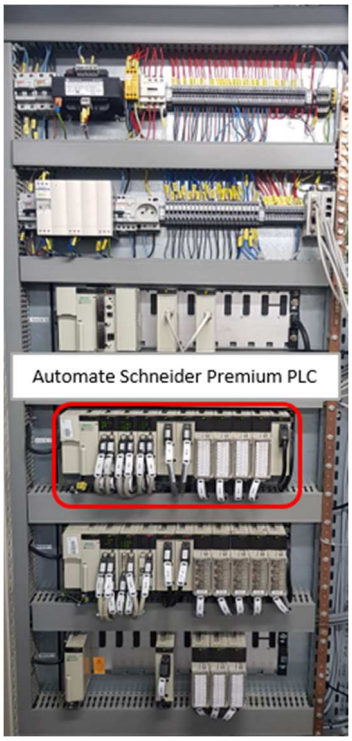

Figure 4 The Programmable Logic Controller (PLC) communicating to Python gateway over OPC.

This experiment is performed on an emulated distribution grid platform, which includes real medium-voltage reduced-scale loads, generators, and a supervisory control and data acquisition system. The reduced-scale grid can be managed by the SCADA system.

The considered communication scenario is a test case that includes an Automate Schneider Premium Programmable Logic Controller (PLC – Figure 4) and a Python gateway communicating through a cloud server over Modbus as the communication protocol. Message transmission delay is measured by a network analyzer, Wireshark, for 1 hour and 23 minutes each 0.5s that results in a time series of 10000 observations.

C. Model application

Next, following the application steps, LSTM model is fitted over training data set, and the estimated model is tested Over both training and test data sets. The predicted values are plotted against measured values in Figure 5.

In order to validate the LSTM predictions over the original values, a loss function is plotted for both training and test dataset. MSE is a common loss function used in case of the LSTM prediction. The fast convergence of test and training loss function to an equal value (MSE=0.01) confirms a reliable prediction model. Related training loss function is shown in Figure 6.

In this paper tuning of the LSTM parameters (i.e., number of epochs, batch size, number of neurons) is made to maximize the fit from the training and test MSE line plots viewpoint (Fig. 6). One epoch means the entire data pass through LSTM only once; an epoch is divided into several batches. Epoch number is selected as 10, 100, 200, 1000, 2000, whereas batch size is 8 and a single neuron is used. Best result is for 100 epochs, as afterwards the MSE decrease saturates. Batch size of 8, 16 and 32 are further tested and training MSE stabilizes sooner if batch size is 16.

Forecast series is not perfectly adjusted over the original one. To find the accuracy of LSTM forecasting model, Mean Absolut Percentage Error (MAPE) is calculated. MPAE is obtained as follows:

$$MAPE = {}^{100}/_{n} \sum_{t=1}^{n} \left| \frac{y_t - \hat{y}_t}{y_t} \right| \times 100, \quad (8)$$

where y_t is the actual value and $\hat{y}_t$ is the forecast value. The forecasting model is excellent if $MAPE < 10\%$, and it is good if $10\% \leq MAPE < 20\%$ [19].

The corresponding MAPE of the predicted series equals to 42.63% that can be concluded as an unsatisfactory prediction. So, clearly the forecasting model needs to be improved.

To better identify data distribution, the histogram of the corresponding dataset is illustrated in Figure 7. Histogram includes multiple peaks. It may be concluded that there are three different trajectories for data transmission.

In the non-faulty situation, messages pass through a default trajectory using some servers in the channel. But, if an error occurs (e.g., a server out of service), messages are forwarded using some backup servers as other possible trajectories.

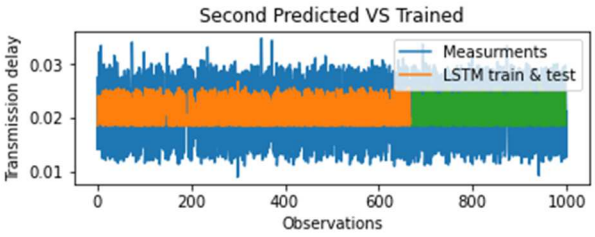

Figure 5 Predictions over the 70% of dataset (training in orange) and predictions over the 30% of dataset (test in green) are compared to the original measured values. $MAPE = 42.63\%$, $MSE = 0.01$.

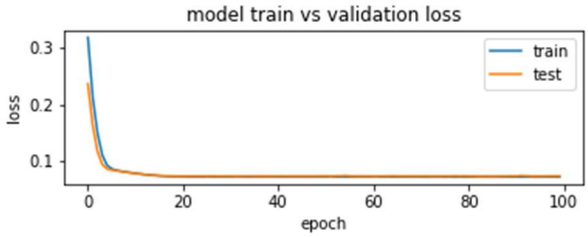

Figure 6 Training set MSE converges to an equal value of test set MSE of 0.01.

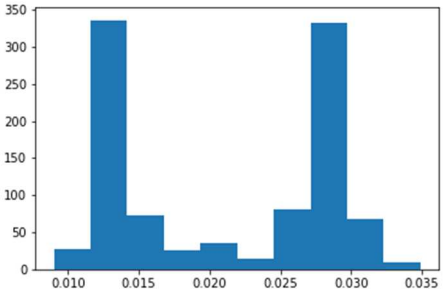

Figure 7 Detection of multiple peaks in histogram of the communication scenario over Modbus.

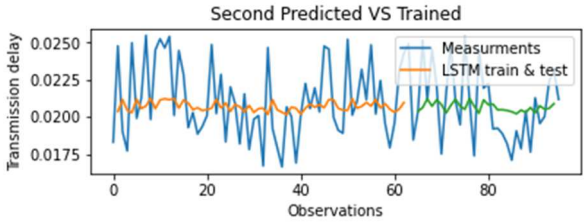

Figure 8 Second-peak-data predictions against the original data. $MAPE = 9.75\%$.

According to this definition, each peak corresponds to a possible trajectory that served messages in this communication scenario.

In order to improve the forecasting LSTM, it may be judicious to model each trajectory dataset separately. To do so, the identified three peaks are separated into three sub-datasets using the mean point between each two consecutive peaks as separation point. Then LSTM is applied to each single peak data. As an example, second peak data is thus identified and the result is represented in Figure 8. The MAPE less than 10% could be considered as an acceptable forecasting model. But, the predictions are not perfectly fitted against the original values in Figure 8. Hence, such an approach is not yet confirmed as the final result and it is still under study to see if the LSTM model can be configured more effectively. Another improvement direction would be use of a more suitable model for high-frequency variations prediction (e.g., ARIMA). Preliminary self-correlation tests are necessary to analyze the degree of randomness in the measured data.

V. Conclusion

In the context of smart grid development, this work aims at bringing legacy systems to the cloud computing. As the interacting applications are vulnerable to the communication delays, it is important to compensate the message transmission delay and its impacts. However, prior to the compensation, there is a need to evaluate the message transmission delay.

Forecasting methods have been selected in this work for compensation. In this case, time-series forecasting models are appropriate to evaluate the transmission delay. Therefore, delay measurement by Wireshark network analyzer is considered as time-series data, and is further identified by an LSTM neural network. Delay is measured over a communication scenario (through Modbus communication protocol) between a PLC and Python gateway. LSTM-based identification is applied through separate steps that results in an unsatisfactory fit ($MAPE = 42.63\%$). To reduce MAPE and obtain a more accurate model, an idea was to apply LSTM on the individual dataset corresponding to each transmission trajectory.

According to the results, it is necessary to see if other LSTM parameters must be regulated to reach a better accuracy. Otherwise, other forecasting models may be applied to gain more accurate fit (*e.g.*, ARMA).

Next step is to validate the proposed forecasting LSTM-based model over a test case that includes two remote platforms through Internet within a distance of several tens of kilometers.

References

[1] V. H. Nguyen, T. L. Nguyen, Q. T. Tran, Y. Bésanger, R. Caire, "Integration of SCADA Services and Power-Hardware-in-the-Loop Technique in Cross-Infrastructure Holistic Tests of Cyber-Physical Energy Systems," *IEEE Transactions on Industry Applications,* vol. 56, no. 6, pp. 7099-7108, Nov.-Dec. 2020.

[2] Y. Wang, T. L. Nguyen, M. H. Syed, "A Distributed Control Scheme of Microgrids in Energy Internet Paradigm and Its Multisite Implementation," *IEEE Transactions on Industrial Informatics,* vol. 17, no. 2, pp. 1141-1153, Feb. 2021.

[3] H. Yang, L. Cheng, X. Ma, "Analyzing Worst-Case Delay Performance of IEC 61850-9-2 Process Bus Networks Using Measurements and Network Calculus," *17th e-Energy: The Eighth International Conference on Future Energy Systems,* p. 12–22, May 2017.

[4] F. Malandra, B. Sansò, "A Markov-Modulated End-to-End Delay Analysis of Large-Scale RF Mesh Networks With Time-Slotted ALOHA and FHSS for Smart Grid Applications," *IEEE Transactions on Wireless Communications,* vol. 17, no. 11, pp. 7116-7127, November 2018.

[5] X. Liu, S. Zhang, X. Zeng, L. Yao, Y. Ding, C. Deng, "Evaluating the network communication delay with WAMS for multi-energy complementary systems," *CSEE Journal of Power and Energy Systems,* vol. 6, no. 2, pp. 402 - 409, June 2020.

[6] Y. Zhu, D. Zhou, M. Sheng, J. Li, Z. Han, "Stochastic Delay Analysis for Satellite Data Relay Networks with Heterogeneous Traffic and Transmission Links," *IEEE Transactions on Wireless Communications,* vol. 20, no. 1, pp. 156 - 170, Jan 2021.

[7] A. Kumari, S. Tanwar, S. Tyagi, N. Kumar, M. S. Obaidat, J. J. P. C. Rodrigues, "Fog Computing for Smart Grid Systems in the 5G Environment: Challenges and Solutions," *IEEE Wireless Communications,* vol. 26, no. 3, pp. 47-53, June 2019.

[8] Y. Long, Z. Lu, H. Shen, "Composable worst-case delay bound analysis using network calculus," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 37, no. 3, pp. 705-709, March 2018.

[9] M. Ateeq, F. Ishmanov, M. K. Afzal, M. Naeem, "Predicting Delay in IoT Using Deep Learning: A Multiparametric Approach," *IEEE Access,* vol. 7, pp. 62022-62031, 14 May 2019.

[10] T. j. Hastie, R. J. Tibshirani, Generalized Additive Models, vol. 43, CRC press, 1990.

[11] S. Hochreiter, J. Schmidhuber, "Long Short-Term Memory. Neural Comput," vol. 9, no. 8, p. 1735–1780, 15 November 1997.

[12] C. Olah, "Understanding LSTM Networks," 27 August 2015. [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[13] Aditi Mittal, "Understanding RNN and LSTM," 12 October 2019. [Online]. Available: https://aditi-mittal.medium.com/understanding-rnn-and-lstm-f7cdf6dfc14e.

[14] G. Rossum, "Python tutorial," Amsterdam, The Netherlands, 1995.

[15] S. Van Der Walt, S. C. Colbert, G. Varoquaux, "The NumPy Array: A Structure for Efficient Numerical Computation," *IEEE Computing in Science & Engineering,* vol. 13, no. 2, pp. 22-30, March-April 2011.

[16] W. McKinney, "Data structures for statistical computing in Python," *In Proceedings of the 9th Python in Science,* vol. 445, pp. 51-56, June 2010.

[17] J. D. Hunter, "Matplotlib: A 2D graphics environment," *IEEE Computing in Science & Engineering,* vol. 9, no. 3, pp. 90-95, 18 June 2007.

[18] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, et al., "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," *Distributed, Parallel, and Cluster Computing, Machine Learning,* no. 2, March 2016.

[19] M., Gilliland, The business forecasting deal: Exposing myths, eliminating bad practices, providing practical solutions, vol. 27, J. W. &. Sons, Ed., 2010.