Real-Time Transient Stability Assessment Based on Deep Recurrent Neural Network

Le ZHENG, Wei HU*
State Key Lab of Power Systems
Department of Electrical Engineering, Tsinghua University
Beijing, 100084, China
*huwei@mail.tsinghua.edu.cn

Kaiyuan HOU, Xingwei XU, Guanghui SHAO
Dispatch and Communication Center,
Northeast Branch, State Grid Cooperation of China
Shenyang, 110180, China

***Abstract*—Real-time transient stability assessment (TSA) is conducted in a data driven framework, by considering the temporal relations of the predictors using recurrent neural network (RNN) with long short-term memory (LSTM) units. Specifically, the TSA problem is converted to approximate the stability boundary based on the fact that the disturbed system is stable if and only if the initial post-fault state is in the region of attraction in state space. Since the *state* variables are difficult to monitor in real world applications, multiple time-step *algebraic* variables (power flows and voltages of the transmission network, *i.e.* P, Q, V, θ) are utilized to approximate the state space by using a LSTM-based model to capture the long-term dependencies along the time steps. The proposed scheme is illustrated on the IEEE 39-bus test system and gets remarkably better testing results compared with a SVM benchmark. The proposed method has shown that involving power system stability domain knowledge is significant to data driven TSA analysis.**

***Index Terms*—Transient stability assessment (TSA), recurrent neural network (RNN), long short-term memory (LSTM).**

I. INTRODUCTION

The ability to remain stable when encountering a large disturbance is vital to enhance power system security [1]. Effective transient stability assessment (TSA) can provide system operators with insightful information on stability statuses and causes under various contingencies. There are basically two main methods for TSA: the time domain simulation and the energy-based direct method [2]. Time domain simulation is time consuming so that it is not suitable for real-time application. As for energy-based direct method, it is very difficult to build energy functions for complex systems. Therefore, traditional methods can no longer satisfy the requirement of online TSA for large-scale power systems.

Since the 1980s, the development of advanced data acquisition techniques and data mining algorithms introduces the possibilities of using data driven approaches to solve real-time transient stability assessment and decision making problems. After offline supervised training, stability status is classified using the system profiles after the clearance of disturbance. Among all the classification techniques, artificial neural network (ANN) has always been one of the most popular algorithms [3]- [7]. This is because well designed and trained ANNs are extremely good non-linear function approximators and representation learners. Generally, ANNs can be used as classifiers to predict binary system stability status (stable/ unstable) [3], [5], [7]. The other common method is to use ANNs in regression tasks to predict continuous indices, such as transient energy margin [4], generators' angles and angular velocities [6]. The predicted indices are then used to judge the stability status.

In this paper, a deep recurrent neural network (RNN) based TSA schema is proposed. In Section II the problem is described and the advantages of using RNN are presented. In Section III, the LSTM-based methodology is proposed. Test results are presented and discussed in Section IV. Section V concludes the work.

II. PROBLEM FORMULATION

A. Data Driven Transient Stability Assessment

For a data driven task, there are naturally three important elements, *i.e.* the input, the algorithm, and the output.

1) Input: In previous works, the inputs are mainly state variables [5]-[7] or algebraic variables [4], [8]. In real world applications, it is not easy to monitor state variables (like rotor angle) so that this paper records the algebraic variables and feeds them into the classifier, including active power P, reactive power Q, voltage magnitude V and phase angle θ.

Another problem is that how to choose proper time period (pre-fault, on-fault, or post-fault) of the variables. Reference [4], [5], and [8] used initial conditions (pre-fault), while [7] and [9] used the measurements at the time of fault clearance and/or successive several cycles (post-fault). In order to make this clear, we need to recall power system stability definition and theory. As stated in [1], in power system stability analysis we are interested in the region of attraction $R(X_p)$ of a given equilibrium set X_p, namely the set of points in the state space

This work was supported by National Key Research and Development Program of China (2017YFB0902201), and Science and Technology Project of State Grid Corporation of China.

978-1-5386-4950-3/17/$31.00 ©2017 IEEE

with the property that all trajectories initiated at the points will converge to the equilibrium set X_p. In most cases, the equilibrium set is a point, which is called the stable equilibrium point (s.e.p), and the cases correspond to asymptotically stable. In other words, for a specific operating condition under various contingencies, the system is transient stable if and only if the state at the time of fault clearance lies inside the region of attraction in state space. In short, the post-fault variables contain full information about the stability status, and therefore are chosen as inputs in this paper.

2) Algorithm: In this paper, we need to approximate the highly nonlinear and complex attractive region of the post-fault s.e.p, which is ideal for ANNs. Furthermore, it is important to take the input data's characteristics into account when determining the ANN's structure and connection [10].

As stated above, the region of attraction is a concept in the state space but we can only get access to the algebraic variables in our application. We need to transform the variables in the algebraic space to the state space. As we know, the algebraic and state variables are connected via the differential algebraic equations (DAE) that describe power system dynamics after encountering the disturbance. Take the single machine infinite bus (SMIB) system as an example. The state variables are the generator's rotor angle δ and angular velocity ω, and the algebraic variable is the electromagnetic power P_e. The relation between the state and algebraic variables is as follows.

$$\delta=\sin^{-1}[(P_eX_T)/(E'E_B)] \quad (1)$$

where E' is the internal voltage of the generator, E_B is infinite bus voltage, and X_T is the equivalent transient reactance. The rotor angular velocity is the derivative of the rotor angle, and can be approximated as

$$\omega=(\delta(t_2)-\delta(t_1))/(t_2-t_1) \quad (2)$$

where $\delta(t)$ denotes rotor angle at time t. Thus a point (δ, ω) in the state space can be represented by the point $(P_e(t_1), P_e(t_2))$ in the algebraic-time space. To generalize, we hypothesize that the algebraic variables of consecutive time steps can be used to approximate the corresponding points in the state space. For more complex systems or more detailed models, more time steps are needed. Therefore, the state space stability boundary identification problem has been converted to a temporal relation learning one. Considering this, the RNN is employed to capture the temporal relations within the algebraic variables, taking advantage of its unique hidden layer design. The RNN structure will be introduced in detail in Section II.B.

3) Output: In this paper, we use a binary output, stable or unstable, to indicate the transient stability status.

B. Long Short-Term Memory Unit

In this part we give a brief overview of RNN and its most popular variant, the long short-term memory (LSTM) [11]. RNNs have a long history of applications in various sequence learning tasks [12]. Different from classical ANN, the RNN neurons in every hidden layer have inner connections so that

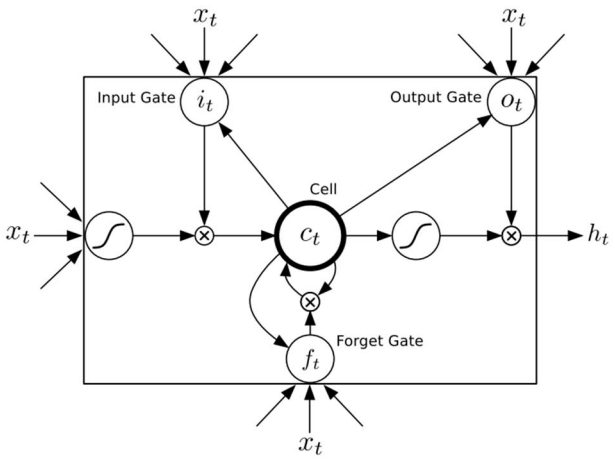

Figure 1. Long short-term memory cell

RNNs can capture information about what has been calculated so far, just like human memory. Unfortunately, for standard RNN architectures, the gradient of a given input on the network output either decays or blows up exponentially as it cycles around the network's recurrent connections, which makes RNNs hard to train. This effect is referred as the vanishing gradient problem [13]- [14].

To address this problem, the approach favored by this paper is the LSTM architecture. A LSTM consists of a set of recurrently connected subnets, known as memory blocks [13]. Each block contains one or more self-connected memory cells and three multiplicative unites- the input, output and forget gates- that provide continuous analogues of write, read and reset operations for the cells [13]. A single LSTM memory cell structure used in this paper is illustrated in Fig. 1, and (3)- (7).

$$i_t=\sigma(W_{xi}\bullet x_t+W_{hi}\bullet h_{t-1}+W_{ci}\bullet c_{t-1}+b_i) \quad (3)$$

$$f_t=\sigma(W_{xf}\bullet x_t+W_{hf}\bullet h_{t-1}+W_{cf}\bullet c_{t-1}+b_f) \quad (4)$$

$$c_t=f_t*c_{t-1}+i_t*\tanh(W_{xc}\bullet x_t+W_{hc}\bullet h_{t-1}+b_c) \quad (5)$$

$$o_t=\sigma(W_{xo}\bullet x_t+W_{ho}\bullet h_{t-1}+W_{co}\bullet c_t+b_o) \quad (6)$$

$$h_t=o_t*\tanh(c_t) \quad (7)$$

where σ is the logistic sigmoid function, and i, f, o, and c are respectively the input gate, forget gate, output gate, and the memory cell, all of which are the same size as the output vector h. Operation • represents matrix multiplication and * is the elementwise multiplication operation. W and b are weights and biases terms of the neurons, which need to be learned through gradient-based backpropagation algorithm [11]. Note that the sigmoid function outputs numbers between zero and one. In this way, the input, output, and forget gates optionally let information through, so as to remove or add information to the cell state and output vector, as illustrated in (5) and (7).

III. PROPOSED METHODOLOGY

A. Generating a Training Database

To predict the stability status of a system in real time, the first step is to produce a training database by performing extensive offline time domain simulations. The uncertainties

978-1-5386-4950-3/17/$31.00 ©2017 IEEE

associated with the location of the faults and fault clearing times are modelled. The faults considered in this paper are three phase faults occurred at the terminals of the transmission lines. The duration of faults is chosen from a scaled normal distribution, from 0.1s to 0.5s. Though the long fault clearing time is unlikely to happen unless the protection system is down, this can generate sufficient unstable cases and patterns for the RNNs to learn from. The length of time domain simulation is 30s so that the stability status can be judged easily at the end of simulation period. If the rotor angle difference of any two generators exceeds 360°, the case is labeled as unstable, otherwise it is stable. The time domain simulations are done by PSAT toolbox on MATLAB platform [14].

As stated above, for each of the cases the response Y_i is 0 (unstable) or 1 (stable). And the predictors are algebraic variables measured from multiple time steps after the fault clearance. Thus the dimension of input feature matrix $\boldsymbol{X}_i$ is $\rho\times(\tau+1)$, where ρ is the number of algebraic variables and τ is the number of successive time steps after the fault clearance. $\boldsymbol{X}_i$ is illustrated in (8), where X represents the vector consisting of the algebraic variables P, Q, V, and θ sequentially in per unit form. Specifically, X_0 is the measurement at the time of fault clearance. There is no feature selection step in this paper since it is believed that more data contain more information, as long as the computation power is sufficient. Note that with proper representation learning step ahead of the training step, the results can be improved to some extent. However, raw data are utilized in this paper for the sake of simplicity.

$$\boldsymbol{X}_i = (X_0^T\ X_1^T\ \cdots\ X_\tau^T) = \begin{pmatrix} x_{01} & \cdots & x_{\tau 1} \\ \vdots & \ddots & \vdots \\ x_{0\rho} & \cdots & x_{\tau\rho} \end{pmatrix} \quad (8)$$

B. Developing a LSTM-Based Classifier

The structure of the LSTM-based classifier is shown in Fig. 2. As we can see from the figure, the input feature matrix $\boldsymbol{X_i}$ is divided into τ+1 vectors, *i.e.* X_0, X_1, … X_τ. Each vector represents the measurements at one single time step and is connected to individual LSTM unit. H is the output vector of the LSTM unit. The τ+1 H vectors are connected sequentially to each other to form a long vector, and then connected to a dense layer activated by the rectified linear units (ReLU) [15]. The dense layer is then connected to another dense layer with two neurons activated by the logistic units, which is employed to predict binary result Y.

Since it is a binary classification problem, logistic unit along with the cross-entropy error cost function is a natural choice to fine tune the LSTM-based classifier. Assume that the stability status of the i-th case is Y_i and the neuron outputs of the prediction layer are respectively y_1 and y_2, the cost function can be written as.

$$J = -\frac{1}{m}\left[\sum_{i=1}^{m} Y_i \log\frac{e^{y_1}}{e^{y_1}+e^{y_2}} + (1-Y_i)\log\frac{e^{y_2}}{e^{y_1}+e^{y_2}}\right] \quad (9)$$

There are two kinds of classification errors, namely false dismissals (FD) and false alarms (FA) [8]. A FD occurs when an unstable case is classified as stable, whereas FAs denote the stable cases deemed as unstable. Considering the conservative requirement of power system operation, the cost of FDs is much higher than that of FAs. Therefore, we use a weighted cross-entropy error cost function (10), to penalize the occurrence of FDs and minimize the FDs rate.

$$\mathrm{J} = -\frac{1}{\mathrm{m}}\left[\sum_{\mathrm{i}=1}^{\mathrm{m}} \mathrm{L}\cdot \mathrm{Y_i} \log\frac{e^{y_1}}{e^{y_1}+e^{y_2}} + (1-\mathrm{Y_i})\log\frac{e^{y_2}}{e^{y_1}+e^{y_2}}\right] \quad (10)$$

where L is the weight, which takes effect when a Y_i=0 is classified as Y_i=1.

The hyper-parameters of the model include the weight, the cell dimension in a LSTM unit, and the number of neurons in the full connection layer. They are determined in a cross validation manner [10]. The LSTM training algorithm is very well documented in the past publications [11], [16], and will not be discussed in detail here. Thanks to the development of GPU parallel computing technique, the training time has been massively shortened.

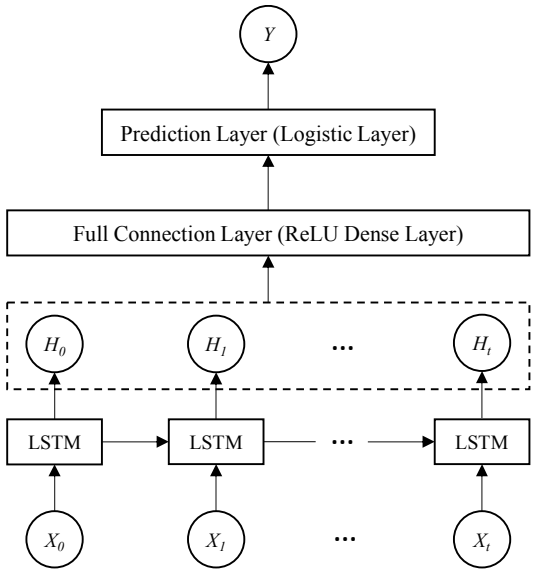

Figure 2. Structure of the LSTM-based classifier

C. Visulizing and Understanding the Gates in LSTMs

To understand how the LSTM-based model performs transient risk prediction, we use visualization tools to analyze the output vectors generated by the model. In particular, the on-and-off behaviors of the input gates, output gates, forget gates, cell states, and the output vectors in the model are examined and analyzed. This profile reveals how the model extracts useful information from the input features and how much information have been delivered.

IV. Case Study and Discussion

The proposed method is applied on transient stability data collected from a IEEE 10-machine 39-bus test system. Three-phase short circuit to ground faults applied at either of the two terminal buses of the thirty transmission lines (except the lines that connect generators to the grid) are considered. A total of 14,000 cases were generated. 3,800 stable and 3,800 unstable cases were randomly selected to form the training dataset. After the model was trained and tuned, it was tested on the testing dataset consisting of 1,300 stable and 1,300 unstable cases. The

978-1-5386-4950-3/17/$31.00 ©2017 IEEE

data dimension for any time step is 169, including 46 active powers, 46 reactive powers, 39 voltage magnitudes, and 38 voltage phase angles (note that there is one reference voltage phase angle). From the time of fault clearance, the data were stored every 10ms (general PMU sampling resolution in China) till 60ms. Therefore, for each observation the dimension of the input feature matrix is 169×7.

A. Classification Performance

A polynomial kernel SVM [17] has been trained on the same dataset and the testing result is used as benchmarks. The LSTM-based model has been trained with normal cross-entropy error cost function (9) and the proposed weighted cross-entropy error cost function (10), respectively. All the optimal hyper-parameters are determined by 5-fold cross validation and grid search. The optimal weight L for the cost function is 5, the dimension of the LSTM memory cell is 256, and the number of neurons in the full connection layer is 500. The model is trained on batches of size 100 [18]. The weights and biases of the LSTM gates and the full connection layer are updated every iteration. Dropout technique is utilized to avoid overfitting [19].

The test results of the three approaches are presented in Table I. It can be seen that the LSTM-based methods outperform the SVM method. The results also suggest that the FD error rate of the model with the weighted cost function is much smaller than that with the normal cost function, though it performs a bit worse considering the overall error rate.

TABLE I. MISCLASSIFICATION ERROR RATE OF THE TESTS

	FA	**FD**	**Overall**
LSTM with weighted cost function	0.66%	0.29%	0.48%
LSTM with normal cost function	0.12%	0.67%	0.39%
Polynomial kernel SVM	4.32%	4.43%	4.37%

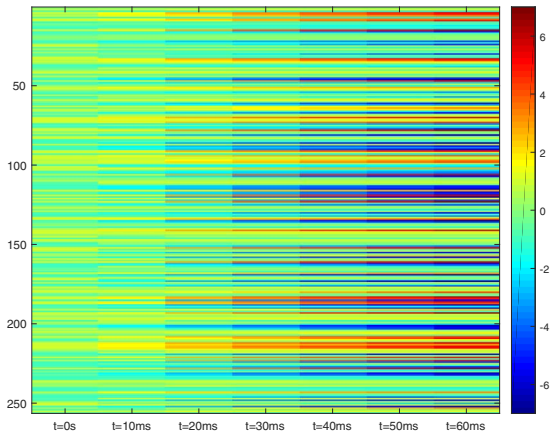

Figure 3. Cell states values $c(t)$ along the time series

B. LSTM Visualization

We now proceed to perform a comprehensive analysis by visualizing the trained LSTM-based model [20]. We would like to examine how the information in the input matrix $\boldsymbol{X}$ is sequentially extracted and embedded into the output vector H over time by the model.

1) Information accumulation: First, we examine the evolution of the cell states and the output vectors. Specifically, an arbitrary case is chosen from the testing dataset and fed into the trained model. Activations of the cell states and the output vectors are shown in Fig. 3 and Fig. 4, respectively. The vertical axis is the memory cell index from 1 to 256, and the horizontal axis is the time step index from left (t=0s) to right (t=60ms). The color codes show activation values: green indicates zeros, while red or blue denotes large absolute values.

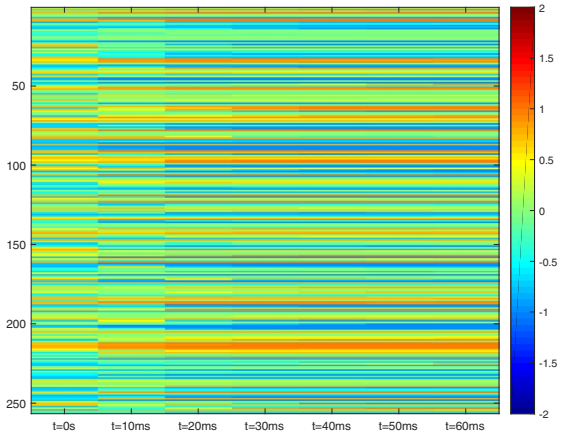

Figure 4. Output vector values $h(t)$ along the time series

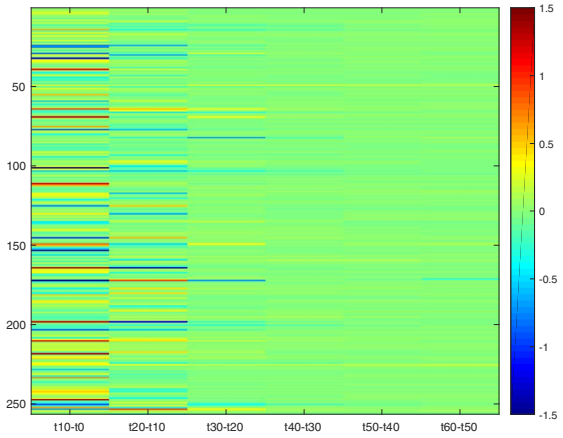

Figure 5. Output vector value variants $\Delta h(t)$ along the time series

From Fig. 3-4, it is clear to see that $c(t)$ and $h(t)$ are evolving over time. Valuable information about the stability boundary is gradually absorbed into $c(t)$ and $h(t)$, so that the information in these two vectors becomes richer as the absolute values grows over time.

2) Information extraction: Second, the information gain of the output vectors is analyzed. Specifically, the variant of adjacent $h(t)$ is computed and shown in Fig. 5. The vertical axis

978-1-5386-4950-3/17/$31.00 ©2017 IEEE

is identical to that in Fig. 3 and 4, while the horizontal axis denotes the difference between adjacent two output vectors (so there are 6 columns instead of 7). For example, the first column in Fig. 5 is shown in (11), and so on and so forth.

$$\Delta h(t10 - t0) = h(t = 10ms) - h(t = 0s) \quad (11)$$

The core idea behind this is that, whenever there is a large change in $h(t)$ we assume important information have been detected by the model. It can be concluded from Fig. 5 that the first three Δh are much larger than the rest. It indicates that the measurements from the first four time steps, *i.e.* X_0, X_1, X_2, and X_3 play a key role in the classification process. Based on this observation, another experiment using the first four time steps data has been conducted. The result is shown in Table II. It is clear that these two experiments leads to similar testing results, indicating that four time steps data are sufficient to provide a good transient stability prediction.

TABLE II. MISCLASSIFICATION ERROR RATE OF THE TESTS

	FA	**FD**	**Overall**
LSTM using 4 time steps data	0.69%	0.33%	0.51%
LSTM using 7 time steps data	0.66%	0.29%	0.48%

On the other hand, the less time steps the model needs, the more reaction time the system operators have. Visualizing the LSTM gates and output can help understand the information evolution along the time series and determine the optimal time steps the model needs. As for the test case, the assessment results can be achieved 30ms (1.5 cycles) after the fault clearance. Compared with the results in [9] (0.5s, or 30 cycles data are needed to make the accuracy above 99.0%), the proposed method can give the system operators and automatic control software more time to react to emergency situations.

V. CONCLUSIONS

We have proposed a data driven method for the problem of transient stability assessment in this paper. Begin with transient stability definition and theory, we have determined the input feature form. In order to model the temporal relations within the predictors, a LSTM-based binary classifier is designed. Then illustrative experiments are conducted and it has been demonstrated that the LSTM-based method achieves remarkably better results compared with a SVM benchmark. By visualizing the gates and memory cells of the LSTM units, the information flow along the time series is analyzed. The results also provide an insight into the possibilities of involving domain knowledge in machine learning process.

ACKNOWLEDGMENT

The authors gratefully thank Hamid Palangi about the discussion on the LSTM visualization part. We thank Yifan Zhou for carefully reviewing the paper and for the continuous support given to us along the past years.

REFERENCES

[1] IEEE/CIGRE joint task force on stability terms and definitions, "Definition and classification of power system stability," IEEE Trans. Power Syst., vol. 19, no. 2, pp. 1387-1401, May 2004.

[2] P. Kundur, *Power System Stability and Control*, 2nd ed. New York, NY, USA: McGraw-Hill, 1994.

[3] Q. Zhou, J. Davidson, and A. A. Fouad, "Application of artificial neural networks in power system security and vulnerability assessment," *IEEE Trans. Power Syst.,* vol. 9, no. 1, pp. 525-532, Feb. 1994.

[4] Y. Mansour, E. Vaahedi, and M. A. El-Sharkawi, "Large scale dynamic security screening and ranking using neural networks," *IEEE Trans. Power Syst.,* vol. 12, pp. 954-960, May 1997.

[5] A. Al-Masri, M. Kadir, H. Hizam, and N. Mariun, "A novel implementation for generator rotor angle stability prediction using an adaptive artificial neural network application for dynamic security assessment," *IEEE Trans. Power Syst.,* vol. 28, no. 3, pp. 2516-2525, Aug. 2013.

[6] N. Amjady, and S. Majedi, "Transient stability prediction by a hybrid intelligent system," *IEEE Trans. Power Syst.,* vol. 22, no. 3, pp. 1275-1283, Aug. 2007.

[7] A. G. Bahbah, and A. A. Girgis, "New method for generators' angles and angular velocities prediction for transient stability assessment of multimachine power systems using recurrent artificial neural network," *IEEE Trans. Power Syst.,* vol. 19, no. 2, pp. 1015-1022, May 2004.

[8] J. Lv, M. Pawlak, and U. D. Annakkage, "Prediction of the transient stability boundary using the Lasso," *IEEE Trans. Power Syst.,* vol. 28, no. 1, pp. 281-288, Feb. 2013.

[9] T. Guo, and J. V. Milanovic, "Online identification of power system dynamic signature using PMU measurements and data mining," *IEEE Trans. Power Syst.,* vol. 31, no. 3, pp. 1760-1768, May 2016.

[10] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, 2nd ed. New York, NY, USA: Springer-Verlag, 2009.

[11] S. Hochreiter, and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, 1735-1780, 1997.

[12] A. Karpathy, J. Johnson, and F. Li, "Visulizing and understanding recurrent networks," in *Proc. International Conference on Machine Learning ICML*, 2016.

[13] A. Graves, "Supervised sequence labelling with recurrent neural networks," Ph.D. dissertation, Technische Universität München, München, 2008.

[14] F. Milano. (2008, Feb. 14). *Documentation for PSAT*. (Version2.0.0) [Online] Available: http://faraday1.ucd.ie/psat.html.

[15] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. International Conference on Artificial Intelligence and Statistics*, 2011.

[16] A. Graves. (2014, Jun.) Generating sequences with recurrent neural networks. [Online]. Available: https://arxiv.org/pdf/1308.0850v5.pdf

[17] L. Moulin, A. Alves da Silva, M. El-Sharkawi, and R. Marks II, "Support vector machines for transient stability analysis of large-scale power systems," *IEEE Trans. Power Syst.,* vol. 19, no. 2, pp. 818-825, May 2004.

[18] C. Laurent, G. Pereyra, P. Brakel, Y. Zhang, and Y. Bengio. Batch normalized recurrent neural networks. [Online]. Available: https://arxiv.org/pdf/1510.01378v1.pdf

[19] N. Srivastava, G. Hinto, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research,* vol. 15, no. 1, pp. 1929-1958, 2014.

[20] H. Palangi, L. Deng, Y. Shen, J. Gao, X. He, J. Chen, X. Song, and R. Ward, "Deep sentence embedding using long short-term memory networks: analysis and application to information retrieval," *IEEE/ACM Trans. Audio, Speech, Language Process,* vol. 24, no. 4, pp. 694-707, 2016.