ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/354702020

Calculating Voltages Without Electrical Models: Smart Meter Data and Neural Networks

Conference Paper · September 2021

DOI: 10.1049/icp.2021.1702

CITATIONS
0

READS
521

3 authors:

Vincenzo Bassi
University of Melbourne
3 PUBLICATIONS **3** CITATIONS
SEE PROFILE

Luis(Nando) Ochoa
University of Melbourne
301 PUBLICATIONS **6,697** CITATIONS
SEE PROFILE

Tansu Alpcan
University of Melbourne
252 PUBLICATIONS **6,799** CITATIONS
SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Project Smart Street View project

Project Advanced Planning of PV-Rich Distribution Networks View project

All content following this page was uploaded by Vincenzo Bassi on 20 September 2021.

The user has requested enhancement of the downloaded file.

CALCULATING VOLTAGES WITHOUT ELECTRICAL MODELS: SMART METER DATA AND NEURAL NETWORKS

Vincenzo Bassi[1], Luis F. Ochoa[1,2], Tansu Alpcan[1]*

[1]Department of Electrical and Electronic Engineering, The University of Melbourne, Melbourne, Australia
[2]Department of Electrical and Electronic Engineering, The University of Manchester, Manchester, UK
**vbassiz@ieee.org*

Keywords: LOW VOLTAGE NETWORKS, NEURAL NETWORKS, PHOTOVOLTAIC SYSTEMS, SMART METERS, VOLTAGE CALCULATIONS

Accepted Paper

Abstract

The growing uptake of residential photovoltaic (PV) systems is driving the need for distribution companies to estimate voltage rise issues in low voltage (LV) networks, for both operational and planning purposes. Operationally, voltage calculations can help determining specific settings (e.g., PV curtailment). In planning, voltage calculations can be used to determine the PV hosting capacity of a given LV network. However, voltage calculations are normally based on power flow analyses and, therefore, require detailed three-phase electrical models which are not readily available for most distribution companies. Taking advantage of smart meters, this paper proposes an approach to calculate voltages without electrical models by capturing the nonlinear relationships among the historical data (demand and voltages) and the corresponding LV feeder using a Neural Network (with one hidden layer). Using cross-validation to determine the most suitable hyperparameters and a realistic Australian LV feeder with 31 single-phase customers, results demonstrate that the approach is very promising as it can accurately calculate voltages for unseen PV scenarios. This approach can make it possible for distribution companies to bypass the time-consuming process of producing LV network models and carry out accurate voltage calculations for any type of what-if scenarios involving PV, batteries, electric vehicles, etc.

1. Introduction

The widespread adoption of residential PV systems is causing voltage rise issues on LV networks, driving the need for distribution companies to accurately estimate potential overvoltage problems, from both operational and planning purposes. To perform an accurate assessment of residential PV systems impacts on customer voltages, distribution companies need to carry out voltage calculations. Operationally, voltage calculations can help determining specific control settings to maintain voltages within statutory limits such as PV curtailment. From a planning perspective, voltage calculations can be used to determine the PV hosting capacity of a given LV network. However, voltage calculations are normally based on power flow analyses and, therefore, in the context of residential customers, require detailed three-phase LV network electrical models which are not readily available for most distribution companies around the world.

Although many distribution companies currently count with georeferenced maps of their LV feeders through Geographical Information Systems (GIS), the corresponding electrical models (topology and parameters) are not available in most cases due to data limitations ranging from impedance values to phase connectivity. This creates a significant barrier for distribution companies to accurately assess the impacts of residential PV systems on customers voltages.

Despite the challenges with electrical models of LV networks, thanks to the growing deployment of smart meters, there is an opportunity for distribution companies to exploit the corresponding data available at customer level, i.e., historical measurements of average active power (P in kW), average reactive power (Q in kvar), and average voltage magnitudes (V in V), corresponding to intervals ranging from a few seconds to 30 mins. These values could be used to capture the physics of a given LV network by applying regression methods. If this is successful within an acceptable accuracy, model-free voltage calculations (i.e., without electrical models) could be carried out by specifying P and Q of each customer at a given instant.

Indeed, recent works, such as [1] and [2], investigate such opportunity. In [1], smart meter data of a real LV network is used. Despite the accurate results, due to the absence of the corresponding LV network electrical model, voltage calculations are validated on historical data only and not assessed for active power values that are outside of the historical data (for instance, larger PV penetration levels). Furthermore, the approach is not disclosed and the effect of reactive power over customers voltages is not considered. In [2] emulated data, i.e., running power flows using a full electrical model, at certain points in the network is used to assess the effectiveness of multiple regressors. A two-step regressor is proposed with accurate results. However, the approach is developed and tested on a single-phase balanced medium voltage (MV) network, i.e., does not cater for the unbalanced nature of LV networks.

Taking advantage of smart meters, this paper proposes a model-free voltage calculation approach based on nonlinear regressions, specifically, using Neural Networks (NNs). Thus, a tailored NN with a single hidden layer is trained to capture the nonlinear relationships among the historical single-phase smart meter data (active powers, reactive powers, and voltage magnitudes of all customers) and the corresponding LV feeder. Once the NN is trained, customers voltage calculations (NN outputs) for any type of what-if scenario of customers demand/generation (NN inputs) are enabled. This approach is adapted from a previous work of the authors [3] using cross-validation to determine the most suitable hyperparameters without requiring any specifications from the historical data (e.g., evolution of PV penetration). The effectiveness of the proposed approach is demonstrated on a realistic Australian LV feeder with 31 single-phase customers. To illustrate the accuracy of the proposed approach on unseen PV scenarios, voltage calculations are assessed using significantly higher PV penetration levels than those observed in the historical data used to train the NN.

2. Voltage Calculations

This section presents the relationships among the historical single-phase smart meter data and the studied LV feeder to be captured by the Neural Network (NN). The historical data of a given LV feeder is represented by the data sets $\boldsymbol{P}$, $\boldsymbol{Q}$ and $\boldsymbol{V}$, which contain the values of average active power, reactive power, and voltage magnitudes, respectively, for all customers, collected at regular time intervals (e.g., every 5, 30 mins) across the considered time period (e.g., a month, a year). The dimensions of $\boldsymbol{P}$, $\boldsymbol{Q}$ and $\boldsymbol{V}$ are given by $|T| \times |C|$, where $|T|$ accounts for the total number of time intervals in the considered time period and $|C|$ represents the total number of customers in the studied LV feeder. Using the historical smart meter data ($\boldsymbol{P}$, $\boldsymbol{Q}$ and $\boldsymbol{V}$), a tailored NN is trained to fit the nonlinear relationships among its inputs ($\boldsymbol{P}$ and $\boldsymbol{Q}$) and its outputs ($\boldsymbol{V}$) as presented in (1), where $\boldsymbol{W}$ corresponds to the matrix of trainable parameters of the NN (further details are presented in section 3).

$$\boldsymbol{V} = f_{NN}(\boldsymbol{P}, \boldsymbol{Q}, \boldsymbol{W}) \tag{1}$$

Once the NN is trained, i.e., f_{NN} is captured by defining suitable NN parameters $\boldsymbol{W}$, voltage calculations are enabled for any type of what-if scenarios. A what-if scenario is represented by the data sets $\overline{\boldsymbol{P}}$ and $\overline{\boldsymbol{Q}}$, which contain specific values of active and reactive power, respectively, for all customers. These data sets are used along with the relationship obtained in (1) to calculate all customer voltages $\overline{\boldsymbol{V}}^{calc}$ as shown in (2).

$$\overline{\boldsymbol{V}}^{calc} = f_{NN}(\overline{\boldsymbol{P}}, \overline{\boldsymbol{Q}}, \boldsymbol{W}) \tag{2}$$

3. Neural Networks

This section presents a brief overview of Neural Networks (NNs). A NN corresponds to a mathematical model composed of interconnected processing units called *neurons* (also known as *perceptrons*). The number of neurons along with activation functions (e.g., Tanh, ReLu, SeLu, etc.), learning rates, and epochs, are denoted as hyperparameters. Hyperparameters correspond to external configuration variables that determine the NN structure and its learning process. On the other hand, NN parameters represent internal variables that are the results of the learning process, such as the weight and bias parameters.

A graphical representation of a single neuron is presented in Fig. 1, where $\boldsymbol{X_n}$ with n in N (set of inputs) corresponds to the n-th input vector and $\boldsymbol{Y}$ represents the neuron output vector, w_n with n in N accounts for the corresponding weight parameters, and the filled grey node is to represent an additional input variable that simply store the value of 1, used to introduce the bias parameter (b_o). Note that $|N|$ accounts for the total number of inputs in N. Thus, as shown in Fig. 1, the output of a single neuron corresponds to an activation function (φ) applied over the weighted sum of its inputs shifted by the bias parameter, as presented in (3) [4].

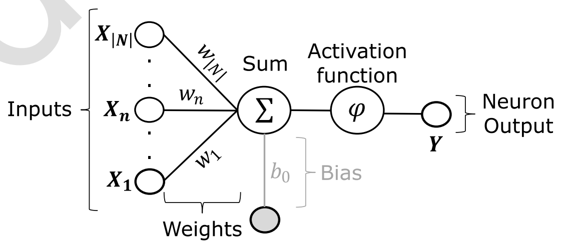

Fig. 1. Single neuron graphical representation

$$\boldsymbol{Y} = \varphi\left(\sum_{n \in N} w_n \boldsymbol{X_n} + b_0\right) \tag{3}$$

A Multilayer Feedforward Neural Network (also known as *multilayer perceptron*) with a single hidden layer is presented in Fig. 2 [5]. This NN is comprised of several fully connected neurons organised in three different layers: input, hidden and output layers. The output of each neuron corresponds to the input of the subsequent neurons as shown in Fig. 2, where $\boldsymbol{P}|\boldsymbol{Q}$ corresponds to the augmented matrix of $\boldsymbol{P}$ and $\boldsymbol{Q}$, $\boldsymbol{P}|\boldsymbol{Q}_{:,n}$ with n in N (set of inputs) accounts for the n-th column vector of the augmented matrix $\boldsymbol{P}|\boldsymbol{Q}$, whereas $\boldsymbol{V}_{:,k}$ with k in K (set of outputs) represents the k-th column vector of $\boldsymbol{V}$. Considering $\boldsymbol{P}|\boldsymbol{Q}_{:,n}$ with n in N as NN inputs and $\boldsymbol{V}_{:,k}$ with k in K as NN outputs, the total number of NN inputs and outputs are equivalent to $2|C|$ and $|C|$, respectively. In this context, it is possible to observe that a NN corresponds to a nonlinear function from a set of input variables $\boldsymbol{P}|\boldsymbol{Q}$ to a set of output variables $\boldsymbol{V}$ adjusted by a set of trainable parameters $\boldsymbol{W}$, as formulated in (4). Note that $\boldsymbol{W}$ accounts for all weight and bias parameters of the NN.

Accepted Paper

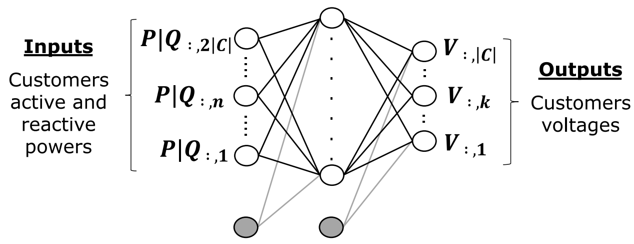

Fig. 2. Multilayer feedforward neural network with a single hidden layer

$$\boldsymbol{V} = f_{NN}(\boldsymbol{P}|\boldsymbol{Q}, \boldsymbol{W}) \quad (4)$$

The process to calculate $\boldsymbol{W}$ is called *training*. The training algorithm adjust the corresponding weight and bias parameters through an iterative procedure in which the error function (e.g., mean squared error) between calculated and real outputs is minimised. This minimisation corresponds to a nonconvex optimisation problem. To solve this problem, gradient based optimisation algorithms along with error backpropagation techniques are commonly used [5]. Once the NN model is trained, the relationship in (1) is captured and, consequently, voltage calculations for any type of what-if scenarios are enabled as shown in (2).

4. Methodology

This section presents the proposed methodology to determine the final Neural Network (NN) model to be used for model-free voltage calculations on what-if scenarios. NNs are trained to extract the relationships presented in (1) for the studied LV feeder. However, the accuracy of the obtained relationships is highly dependent of NN hyperparameters and parameters. To find the most suitable NN model (i.e., hyperparameters and parameters) that allows to accurately capture these relationships, the proposed approach considers three stages: First, the training data set is extracted from the available historical single-phase smart meter data. Then, NN hyperparameters are determined based on problem characteristics along with a K-fold cross-validation scheme. Finally, a NN with the defined hyperparameters is trained to determine all NN parameters and, consequently, the final NN model.

4.1. Training data set

The training data set $\boldsymbol{P_{train}}$, $\boldsymbol{Q_{train}}$, and $\boldsymbol{V_{train}}$ correspond to continuous samples extracted from the historical single-phase smart meter data $\boldsymbol{P}$, $\boldsymbol{Q}$, and $\boldsymbol{V}$, that contains the values of average active power, reactive power, and voltage magnitudes, respectively, for all customers in the studied LV feeder, collected at regular time intervals throughout the considered training period.

4.2. Hyperparameter selection

To select the most suitable NN hyperparameters, the first step is to determine NN hyperparameters that can be obtained directly by problem characteristics. In this context, the input layer dimension corresponds to $2|C|$ (active and reactive powers of all customers), whereas the output layer

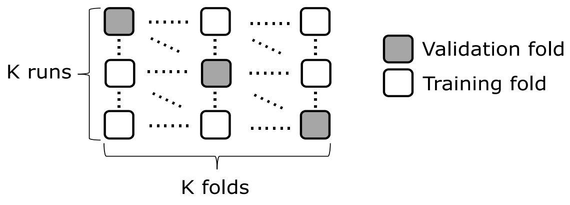

Fig. 3. K-fold cross-validation scheme

dimension is given by $|C|$ (voltages of all customers). Due to the regression nature of the problem, mean squared error (MSE) as error function as well as linear activation function for the output layer are considered. Data sets are scaled to values within the range [0,1] to speed up learning and convergence processes, and the ADAM algorithm [6] is used to solve the nonconvex optimisation problem. Finally, to improve the generalisation capabilities of the NN, i.e., the ability of the NN to perform well on different scenarios than those observed during training, L2 regularisation penalties over the loss function as well as constraints over the norm of the weight and bias parameters of each layer (i.e., $|W_l| \in [0,1]$ where W_l is comprised of all the weight and bias parameters of the l-th layer) are implemented.

Once the previous hyperparameters are defined, a K-fold cross-validation scheme (presented in Fig. 3) is used to determine the remaining hyperparameters: hidden layer neuron number, activation function, learning rate, L2 regularisation factor and epochs. Thus, the training data set ($\boldsymbol{P_{train}}$, $\boldsymbol{Q_{train}}$, and $\boldsymbol{V_{train}}$) is split in k disjoint folds of equal dimensions. As presented in Fig. 3, for each combination of hyperparameters, a NN is trained k independent times, each time a different set of $k-1$ folds are considered to train the NN whereas the remaining fold is used as validation fold to assess NN accuracy at each run (given by the MSE between calculated and actual voltages in the corresponding validation fold). Each NN model is trained using a batch size of 48 points (equivalent to one day length). Hence, the training data is shuffled and divided into disjoint continuous batches of 48 points before each epoch. NN model updates are carried out using every batch throughout each epoch of the training process. Finally, the NN with the best cross-validation accuracy, calculated as the average of the validation accuracies of the k runs, determine the remaining hyperparameters.

4.3. Parameter selection

The last stage is to determine NN parameters. To this end, a NN with all the defined hyperparameters is trained from scratch using the full training data set (all k folds for training). The obtained NN model corresponds to the final NN model ready to be used for model-free voltage calculations on what-if scenarios.

5. Results

This section presents the results of the case study. As case study, model-free voltage calculations are carried out on a realistic three-phase LV feeder (400V line-to-line) with 31

Accepted Paper

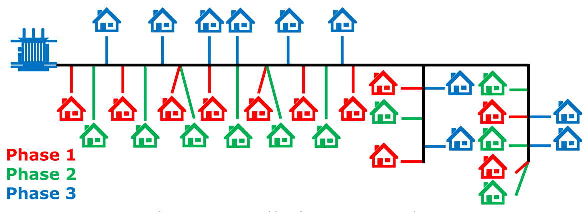

Fig. 4. Realistic LV Feeder

single-phase customers from Victoria, Australia (presented in Fig. 4). The LV feeder is supplied by a transformer of 22kV/433V with the off-load tap changer in its nominal position, i.e., providing a natural boost of approximately 8% which is often seen in Australia. 31 single-phase customers are connected through single-phase service cables, from which 11, 10 and 10 customers are connected to phase 1 (red), phase 2 (green), and phase 3 (blue), respectively.

5.1. Synthetic smart meter data

To illustrate the accuracy of the proposed approach on unseen PV scenarios, voltage calculations are assessed with significantly higher PV penetration levels than those observed in the historical data ($\boldsymbol{P}_{train}$, $\boldsymbol{Q}_{train}$, and $\boldsymbol{V}_{train}$) with which the NN is trained. To emulate this with an adequate correspondence, a synthetic data approach is implemented. Hence, power flow simulations are carried out in OpenDSS [7] using real half-hourly active power demands along with normalised PV generation profiles from Victoria, Australia (from 2016 and 2014, respectively, provided by the Australian distribution company AusNet Services). A random inductive power factor between 0.90 and 0.99 for each customer at each half-hourly interval is used to emulate reactive power demand. Thus, a training data set is created to emulate 3 consecutive winter weeks with 0% of PV penetration in the studied LV feeder, whereas two test data sets (what-if scenarios) are created to emulate 3 consecutive summer weeks with 51.6 % and 100% of PV penetration in the studied LV feeder, i.e., 16 and 31 customers with a 4.5 kW PV system each, respectively.

5.2. Neural network model selection

The algorithm used to train and assess each NN model is implemented in Keras [8] which corresponds to a deep learning library written in Python that runs on top of the machine learning open-source platform Tensorflow [9]. The training data set is used to define suitable hyperparameters and parameters and, therefore, the final NN model, through the following process.

The first step is to define the hyperparameters which are determined by problem characteristics. Thus, the next hyperparameters are defined: 62 as NN input dimension (active and reactive powers of the 31 customers), 31 as NN output dimension (voltages of all customers), linear activation function for the output layer as well as MSE as loss function (regression problem), ADAM as optimiser (robustness [6]), and L2 regularisation along with weight and bias parameters constraints (to enhance NN generalisation capabilities).

Then, with these hyperparameters defined, the remaining hyperparameters need to be determined. To this end, a search is carried out based on a K-fold cross-validation scheme (Fig. 3) splitting the training data set in 3 folds of 1-week length. The search is carried out using several combinations (810 in total) of neuron numbers (from 31 to 310 considering a fixed step size equals to 31), activation functions (Tanh, ReLu, SeLu), learning rates along with L2 regularisation factors ($1x10^{-3}$, $1x10^{-4}$, $1x10^{-5}$), and epochs (500, 1,000, 2,500). Using a batch size of 48 points, the NN with the best cross-validation accuracy (0.0001 V^2) is found with 248 ($31x8$) neurons considering Tanh as activation function, a learning rate of $1x10^{-4}$, a L2 regularisation factor of $1x10^{-5}$, and 1,000 epochs. The full results are presented in Table 1.

Table 1. Neural Network Hyperparameters

Input layer	62	Optimiser	ADAM
Neurons HL	248	Learning rate	$1x10^{-4}$
Act. Func. HL	Tanh	Regularisation	L2 ($1x10^{-5}$)
Output layer	31	Epochs	1,000
Act. Func. OL	Linear	W&B Const.	$\lvert W_l \rvert \in [0,1]$
Loss Func.	MSE	Batch size	48 points

The obtained NN accounts for a total of 23,343 parameters. The final step is to determine the values of these parameters. To this end, a NN with the defined hyperparameters (Table 1) is trained from scratch using the entire training data set (all 3 folds) to train the NN. This NN model define all NN parameters and correspond to the final NN model to be used for model-free voltage calculations.

5.3. Model-free voltage calculations results

The performance of the final NN model is assessed using both test data sets (i.e., 51.6% and 100% of PV penetration). For both cases, time series voltage calculations carried out by the final NN model along with the corresponding test voltage values (actual voltages), and voltage calculations deviation (given by calculated voltages minus test voltage values) are presented in Fig. 5 for the customer with the biggest voltage rise issue due to PV generation (i.e., the customer located at the end of the LV feeder). Voltage calculations overall performance is shown in Table 2, where MSE, average absolute deviation (Av. Dev.), and maximum absolute deviation (Max. Dev.) metrics are presented. Due to the different PV penetration levels considered for the training data set (0%) and the test data sets (51.6% and 100%), voltage calculations performance is presented for solar (06-20 hrs) and nonsolar hours (remaining hours).

Results observed in Fig. 5 and Table 2 demonstrate that the proposed model-free voltage calculations approach achieves a very accurate performance, despite the significant differences in LV feeder PV penetration levels between the training data set (no PV) and the test data sets (51.6% and 100% PV). The very low MSE, Av. Dev., and Max. Dev. values show that the proposed approach is able to perform accurate voltage calculations on unseen PV penetration scenarios, which is critical for operational and planning purposes. It is important to note that the overall

Accepted Paper

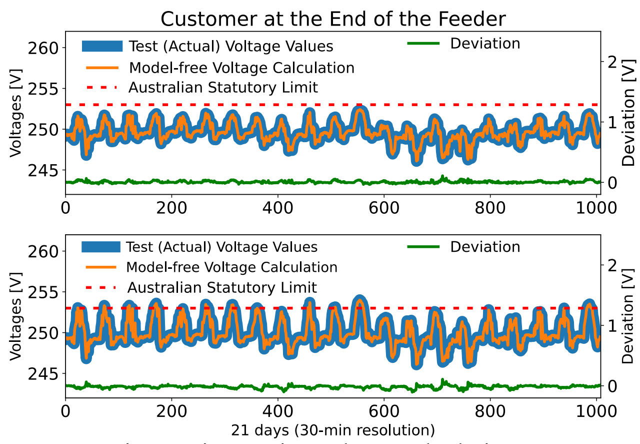

Fig. 5 Time series voltage calculations. (Upper) 51.6% PV Penetration test case, (Lower) 100% of PV Penetration test case

Table 2. Model-free voltage calculations overall results

PV Penetration	Metric	Solar hours	Nonsolar hours	Global
51.6%, (16/31 customers with PV)	MSE. (V^2)	0.0020	0.0002	0.0011
	Av. Dev. (V)	0.0318	0.0074	0.0206
	Max. Dev. (V)	0.2520	0.2234	0.2520
100%, (31/31 customers with PV)	MSE. (V^2)	0.0030	0.0002	0.0017
	Av. Dev. (V)	0.0372	0.0074	0.0236
	Max. Dev. (V)	0.2941	0.2234	0.2941

performance decreases during solar hours. Nevertheless, great accuracy is accomplished in both solar and nonsolar hours. Finally, the average percentage error corresponds to 0.0083% and 0.0094% in the 51.6 and 100% PV penetration test data set, respectively. Very promising results to calculate voltages without the need for LV feeder electrical models and power flow analyses.

6. Conclusions

Detailed three-phase LV networks electrical models (topology and parameters) are not readily available for most distribution companies around the world. This creates a significant barrier for distribution companies to perform power flow-based analysis to accurately assess the impacts of residential PV systems on customer voltages. Taking advantage of smart meters, this paper proposes a model-free voltage calculation approach that captures the nonlinear relationships among the historical single-phase smart meter data (active powers, reactive powers, and voltage magnitudes of all customers) and the corresponding LV feeder through nonlinear regressions using a Neural Network (NN) with a single hidden layer. Thus, for a given LV feeder, a tailored NN is trained to fit the nonlinear relationships among its inputs (active and reactive powers of all customers) and its outputs (voltage magnitudes of all customers). The trained NN enables distribution companies to perform accurate and extremely quick voltage calculations for any kind of what-if scenarios. To find the most suitable NN hyperparameters for the studied LV feeder, a cross-validation scheme is implemented.

As study case, voltage calculations are carried out on a realistic three-phase LV feeder (400 V line-to-line) with 31 single-phase customers from Victoria, Australia. The obtained results demonstrate that the proposed approach can perform very accurate voltage calculations on what-if scenarios consisting of significantly higher PV penetration levels than those observed in the historical data used to train the NN model, all without the need for power flow analyses and LV feeder electrical models, enabling distribution companies to bypass the time-consuming process of producing LV network models and carry out accurate voltage calculations for any type of what-if scenarios involving PV, batteries, electric vehicles, etc. Furthermore, due to the nature of the calculations performed by the NN model (direct equations), the proposed approach represents not only an accurate but also an extremely quick alternative to power flow-based voltage calculations making it suitable for both operational (e.g., determining specific settings such as PV curtailment) and planning analyses (e.g., PV hosting capacity assessment).

Finally, other regression methods (e.g., support vector regression, polynomial fitting, and gaussian process regression) could also perform well and, therefore, need to be investigated. Furthermore, the impacts of the medium voltage network over the proposed approach also need to be assessed.

Accepted Paper

References

[1] R. Pellerej, T. Trouillon, C. Benoit, Q. Garnier, A. Versyp. "Impact of Flexibility on Low Voltage Network's Hosting Capacity – Belgium Experimentation," Proc. 2020 CIRED Workshop, Berlin, Germany, Sept. 2020, p. N° 405.
[2] A. Furlani Bastos, S. Santoso, V. Krishnan and Y. Zhang, "Machine Learning-Based Prediction of Distribution Network Voltage and Sensor Allocation," 2020 IEEE Power and Energy Society General Meeting, Montreal, Canada, August 2020, pp 1 -5.
[3] V. Bassi, L. Ochoa and T. Alpcan, "Model-Free Voltage Calculations for PV-Rich LV Networks: Smart Meter Data and Deep Neural Networks," 2021 IEEE Madrid PowerTech, Madrid, Spain, July 2021, pp. 1-6.
[4] D. J. C. Mackay, "Information Theory, Inference & Learning Algorithms", Cambridge University Press, 2002.
[5] C.M. Bishop, "Pattern recognition and machine learning," New York: Springer, 2006.
[6] D.P. Kingma and J. Lei Ba, "ADAM: A Method for Stochastic Optimization," 3rd Int. Conf. on Learning Representations, San Diego, CA, USA, May 2015
[7] OpenDSS – Open Distribution System Simulator. EPRI. https://www.epri.com/pages/sa/opendss
[8] Chollet, F., & others. (2015). Keras. https://github.com/fchollet/keras
[9] Abadi M., & others (2016). Tensorflow. https://github.com/tensorflow/tensorflow

View publication stats