Bayesian Compressed Deep Learning for State Estimation of Unobservable Power Systems

Rafael Lima
Samsung R&D Institute Brazil
Av. Cambacicas 1200, Campinas-SP, Brazil 13097-104
Email: rafael.goncalves.lima@gmail.com

***Abstract*—In recent years, state-of-the-art Deep Learning (DL)-based modeling has been applied to the problem of state estimation of unobservable electrical distribution systems, with promising results. Unfortunately, the definition and training of these flexible models have been largely heuristic, which may result in oversized neural networks, with computationally inefficient layers. In this work, we apply the method of Bayesian Compression for eliminating spurious redundancies of DL-based State Estimation models. Experimental results in four test networks, including two IEEE Test Case Power Networks, corroborates the benefits of the proposed compression approach for obtaining reduced versions of the models without compromising their performance.**

I. Introduction

State Estimation in Power Distribution Systems is a crucial procedure for their continuous monitoring, along with its consequent reliability and safety of their operation. Its relevance has continuously grown, in a world with increasingly more distributed and real-time operated electrical networks.

A common pitfall in power networks is their unobservability, which is to say that the available set of measurements correspond to uncountably many possible states. This unavailability may result from insufficient number of sensors available across the network, but also from faulty sensing devices, which may output null or unreliable measurement values.

Recently proposed approaches [1], [2] based on neural networks with multiple hidden layers, denominated *"Deep Learning"* (DL), have shown promising results on the problem of State Estimation of unobservable electrical networks, taking advantage of the expressiveness and *Universal Approximation* [3] capabilities of such models for approximating joint distributions of partial measurements' and network states' values.

Unfortunately, much of DL modeling is based on manual tuning, rules of thumb, or just plain trial and error. This lack of rigour in Power System state modeling step is likely to result in oversized neural networks, with weights which exhibit redundancy with respect to the remaining of the model parameters.

For dealing with this accidental model oversizing, several approaches of Model compression were proposed, which act on the reducing the complexity of the model by either pruning (i.e., discarding redundant or irrelevant weights) or bit precision reduction, which reduces the number of float digits of the weights as a way of lowering the number of floating point operations involved in the model's computations. Among them, the so-called *Bayesian Compression* (BC) [4] uses a variational approach for simultaneous model pruning and bit precision reduction, which we will adapt here for the case of state estimation of unobservable Power Systems.

In the following, we describe the steps of our Bayesian Compressed Deep Learning-based State Estimation in Unobservable Power Systems (BCDL-SEUPS), as well as the main improvements with respect to the current works involving DL modeling of Power Systems, mainly introduced in [1] [2]. The main differences are:

1) The Distribution Learning of typical injection/consumption values from historical data, here pursued with standardized implementation of Gaussian Mixture Models (GMM), is done in a systematic way, through the Bayesian Information Criterion (BIC) for Model Regularization
2) The DL modeling step for state estimation is followed by a posterior one, which compresses the fitted model through the simultaneous weight pruning and bit precision reduction procedures of the Bayesian Compression method

The efficacy of proposed improvements is validated through reproducible experiments with four example networks, including two IEEE Test Case networks (14s and 57). Implementation Code and Data will be made publicly available.

II. Proposed Approach

Let be a Power Distribution System $\mathcal{S}$ with N nodes. We denote the State $\mathbf{Y} \in \mathbb{R}_N^+$ of the System $\mathcal{S}$ as a N-sized positive-valued vector

$$\mathbf{Y} = \{Y_1, Y_2, ..., Y_N\} \tag{1}$$

The generalization for node states with three-phase voltages and angle components is left for future work.

This state of the system is affected by the architecture of the network, as well as the power injection or consumption acting along each node of the network. Here, we summarize the injection and consumption of power in the form of a N-sized real-valued Input vector $\mathbf{X} \in \mathbb{R}_N$.

$$\mathbf{X} = \{X_1, X_2, ..., X_N\} \tag{2}$$

By definition, a negative entry X_i corresponds to a resulting power consumption at the i-th node, while a positive value corresponds to power injection.

Associated with the assumed unobservable system, we have a M-sized Partial Measurement vector $\mathbf{Z} \in \mathbb{R}_M^+$, with

$$M = 2\lfloor \kappa N \rceil, \text{ s.t. } \kappa \in (0, 1 - \frac{1}{2N}) \tag{3}$$

where the operator $\lfloor \cdot \rceil$ corresponds to rounding the operand to the nearest integer. In the present work, the M entries of $\mathbf{Z}$ correspond to a M-sized randomly sampled subset of indexes from 1 to N.

A. Deep Learning for State Estimation in Power Systems

Inspired by recent works [1] [2], we propose to model the response of the distribution system through a learning-based strategy, which consists of a model of the conditional probability of the network state given its partial measurement input vector: $p(Y|Z)$

The DL approach consists of fitting an approximate model

$$\tilde{p}_\theta(\mathbf{Y}|\mathbf{Z}) \approx p(Y|Z), \tag{4}$$

in the form of a flexible enough neural network, to a dataset $\mathcal{D}$ consisting of D input-output example pairs

$$\mathcal{D} = \{\mathbf{Z}_k, \mathbf{Y}_k\}_{k=1}^{D}, \tag{5}$$

through the minimization of a corresponding loss function, s.a. Mean Absolute Error (MAE) or Mean Squared Error (MSE). The vector θ denotes the weights of the neural network.

These examples are generated by applying different injection values of $\mathbf{X}_k$ to the network and running the power flow equations of the networks, for obtaining the corresponding values of $\mathbf{Z}_k$ and $\mathbf{Y}_k$. For bringing the network model close to real-world operation values, we fit a probabilistic model $p(X)$ of injection values with historical data, such as those of publicly available datasets of household energy consumption across time. More details are given in the Experimental section.

Each sampled value $\mathbf{X}_k$ is then applied to the model of the network, and the corresponding input-output pair $\{\mathbf{Z}_k, \mathbf{Y}_k\}$ is obtained through running the corresponding network power flow equations. The procedure is repeated until the D samples are obtained.

B. Bayesian Compression for Deep Learning

Deep Learning has recently enjoyed widespread acclaim in a number of tasks, such as Image Recognition and Speech Translation, due to its flexibility of modeling which allows the proposed models to accurately capture nuances of high-dimensional and large-sized datasets.

However, well-known drawbacks of fitting these large variance models are the risk of overfitting and the lack of interpretability. Besides, the modeling procedure has very few theoretical guarantees regarding the choices of model architecture and optimization algorithm. Most of the decisions are heuristic and based on domain expertise.

This lack of strict guidelines may lead to choices of architecture, here implied by the model weights θ, which result in a good portion of the model exerting redundant influence in the input-output mapping and a consequential computational inefficiency of the model's layers.

For circumventing these model's inefficiencies and redundancies, the DL has successively proposed compression techniques for the models, which aim to optimize the computational cost of the model operation while preserving its accuracy and overall performance indicators through two main strategies:

1) Pruning layers, for removing redundant weights;
2) Reducing the floating bit precision of the weights to the minimum required to maintain the accuracy of the model's computation.

The method denoted *Bayesian Compression* [4]achieves both strategies by using a variational inference to the probabilistic distribution of weights. By assuming that the weights of the neural network can be represented by a probabilistic distribution with mean zero:

$$\theta \sim \mathcal{N}(\theta; 0, \eta^2) \tag{6}$$

where the parameter η, associated to the variance, which implicitly controls the sparsity of the weights, is also represented by a parametric probability distribution

$$\eta \sim p(\eta) \tag{7}$$

which can be defined, for example, as the hyperparameter free log-uniform prior or the half-Cauchy prior.

It is then possible to define an approximate joint distribution $q_\phi(\theta, \eta)$ so as to maximize a lower bound on the loglikelihood of the observed data $\mathcal{D}$ regularized by this variational approximation

$$\begin{aligned}\mathcal{L}(\phi) &= \mathbb{E}_{q_\phi(\theta,\eta)}\left[\log p(\mathcal{D}|\theta)\right] - \mathbb{E}_{q_\phi(\eta)}\left[KL(p(\theta,\eta)||q_\phi(\theta,\eta))\right] \\ &= \mathbb{E}_{q_\phi(\theta,\eta)}\left[\log p(\mathcal{D}|\theta)\right] - \mathbb{E}_{q_\phi(\eta)}\left[KL(p(\theta|\eta)||q_\phi(\theta|\eta))\right] \\ &- KL(p(\eta)||q_\phi(\eta))\end{aligned} \tag{8}$$

where $KL(\cdot||\cdot)$ refers to the Kullback-Leibler Divergence

$$KL(p(\cdot)||q(\cdot)) = \mathbb{E}_{p(\cdot)}\left[\log\left(\frac{p(\cdot)}{q(\cdot)}\right)\right] \tag{9}$$

By assuming parametric forms for $p(\eta)$, the second and third terms of the right side of Equation 8 can be computed in closed-form. From the final resulting variance of the approximated distribution, it is possible to infer the unit round off required to represent the weights.

The steps of our full Bayesian Compressed DL-based State Estimation method are described in Figure 1.

III. Experiments

We evaluated the performance of the proposed compressed estimation pipeline on four network test cases. The first two, with 3 and 6 buses, are implemented using the PyPSA library, including the power flow simulations, and the last two are IEEE standard network test cases (14s and 57). Diagrams of the IEEE networks are shown in Figure 2. We used the standardized versions of both IEEE networks available, as long

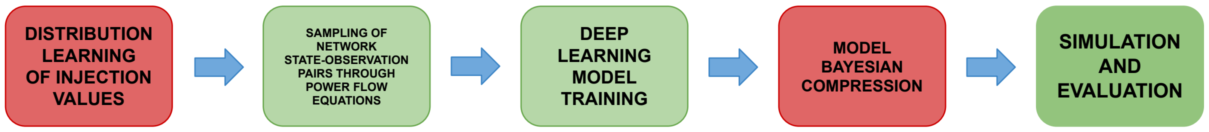
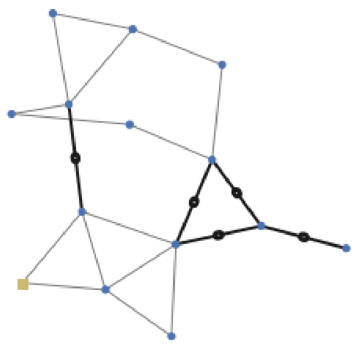

DISTRIBUTION LEARNING OF INJECTION VALUES
SAMPLING OF NETWORK STATE-OBSERVATION PAIRS THROUGH POWER FLOW EQUATIONS
DEEP LEARNING MODEL TRAINING
MODEL BAYESIAN COMPRESSION
SIMULATION AND EVALUATION

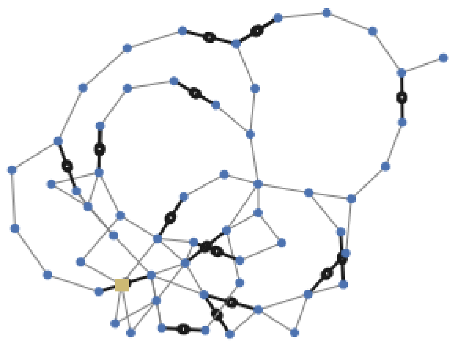
Fig. 1: Summary of the steps of our Bayesian Compressed DL-based State Estimation method. The steps in red correspond to the ones which were added or modified in our current procedure, when comparing with other DL-based methods.

as the power flow equation simulations, through the Python *pandapower* library. Source code was made available [1].

Our goal was to evaluate the performance of the compressed estimation pipeline across different values of κ and also different network architectures. For that purpose, we used four-layered Multilayer Perceptrons (MLPs) and varied the number of neurons in the two hidden layers.



(a) IEEE Network Test Case 14s



(b) IEEE Network Test Case 57

Fig. 2: Schematic diagrams of the test case networks, indicating transformers (black), buses (blue), lines (gray) and external grid (yellow).

A. Distribution Learning of Network Injection Values

For distribution learning of typical injection values, the household consumption data from Ausgrid Solar home electricity data [5] was used. The Gaussian Mixture Model (GMM) was chosen as the probabilistic model, due to its modeling flexibility. The *scikit-learn* implementation of GMM was fitted in the data from 01/Jul/2011 to 30/Jun/2012 and tested in the data from 01/Jul/2012 to 30/Jun/2013.

[1]https://github.com/rafael-glima/BCDL_SEEEDN

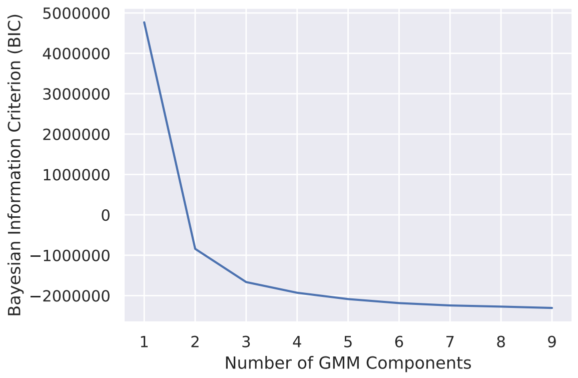


Fig. 3: Evaluation of Bayesian Information Criterion (BIC) as a function of the number of components of the Gaussian Mixture Model (GMM).

For choosing the appropriate number of gaussian components, we used the Bayesian Information Criterion (BIC) [6]:

$$BIC = k\ln(n) - 2\ln(\hat{L}), \tag{10}$$

where:

- k: Number of parameters of the GMM;
- n: Number of data points
- $\hat{L}$: The maximized value of the likelihood of the GMM

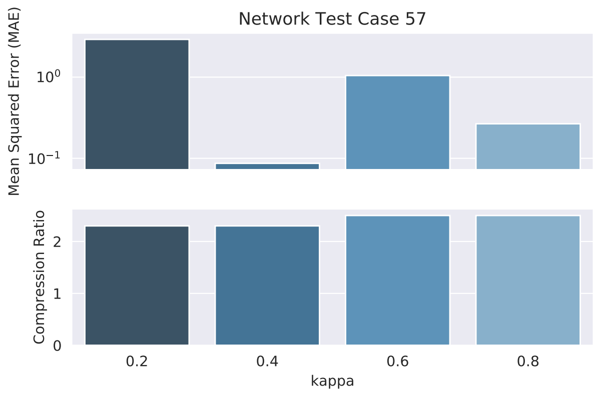
As it can be seen in Figure 3 raising the number of components above 5 has little effect in the overall BIC performance, and so we set number of gaussians for the GMM as 5.

Initially , it was intended to model solar power consumption separately from the other energy sources, but solar data readings were absent in the dataset. Thus, only a single GMM model was used for all energy consumption readings.

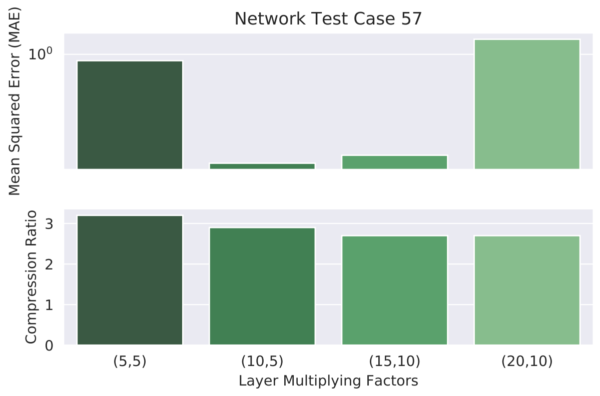
B. PyPSA Network Test Cases

The first network is a minimal test case, composed of 3 buses, connected by lines. It was implemented with the Python library PyPSA [7].

The second network is a publicly implementation of the opf-storage-hvdc example, which is composed of two three-node AC networks connected by 2 point-to-point DC links

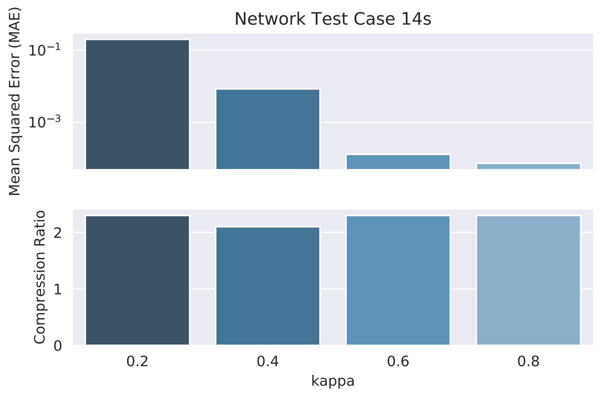


(a) Validation MSE and Compression Ratio as a function of parameter κ. The model architecture was fixed to $(10, 5)$.

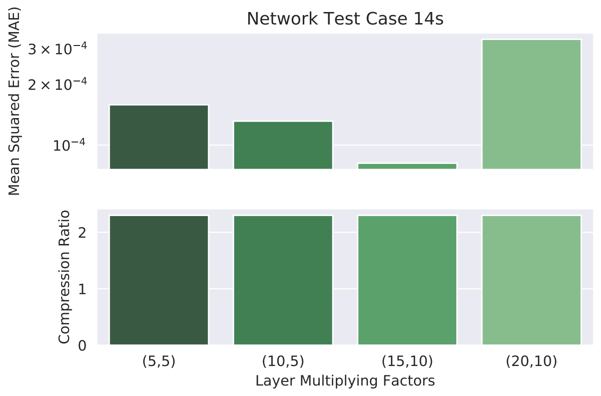


(b) Validation MSE and Compression Ratio as a function of Network Architecture (Multiplying Factors for Hidden Layers units)

Fig. 4: DL Modeling and Compression performance results for IEEE Network Test Case 57



(a) Validation MSE and Compression Ratio as a function of parameter κ. The model architecture was fixed to $(10, 5)$.



(b) Validation MSE and Compression Ratio as a function of Network Architecture (Multiplying Factors for Hidden Layers units)

Fig. 5: DL Modeling and Compression performance results for IEEE Network Test Case 14s.

[2]. We chose four different combinations of multiples of the dimension of the partial measurement vector for the number of weights in each layer:

$$\{(5,5),(10,5),(15,10),(20,10)\}, \tag{11}$$

in which (x, y) means that the first hidden layer is going to have $x * M$ weights, while the second hidden layer has $y * M$ weights.

In this case, the power flow equations were run over snapshots we considered on the first 12 hours in 2015. We obtained a total of 100 samples and used them to fit the discriminative DL model across 50 epochs. For the opf-storage-hvdc network, the κ was varied between $\{0.2, 0.4, 0.6, 0.8\}$, while the smaller 3-bus network had κ varied among $\{0.4, 0.6, 0.8, 1.0\}$, since the vector with only 20 % of its 3 node measurements would be an empty one.

It can be seen that the compression ratio stays above 2 for a wide range of network architectures and observable nodes' ratio. For the 6-bus network, increasing the complexity of the MLP model, and also the observability parameter κ, decreases the resulting MSE in the test set. For the smaller 3-bus network, increasing the complexity of the MLP decreases its test set performance, which indicates possible overfitting.

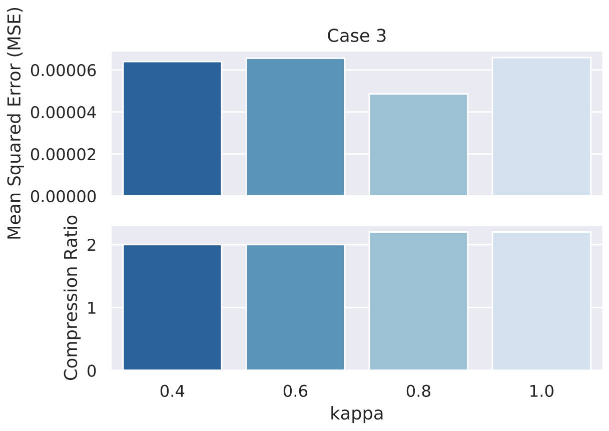
[2]The full network description can be found at https://github.com/PyPSA/PyPSA/blob/master/examples/opf-storage-hvdc/export_opf-storage_network.py

C. *IEEE Test Case 14s*

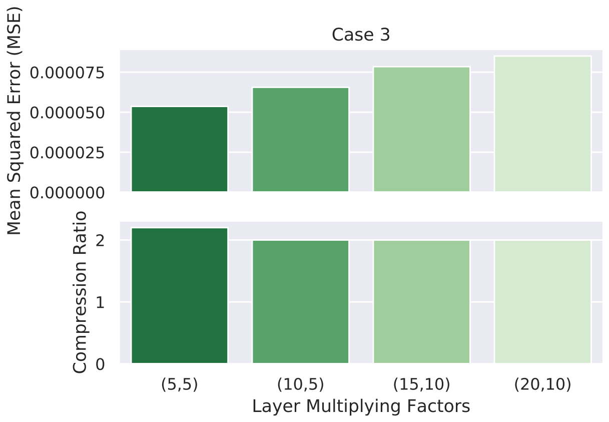
We initially used sampled injection values from the GMM to set the states of the network buses and run the power flow equations. Then, we obtained a partial measurement vector composed of a given percent, set as κ of the total number of buses. We obtained a total of 100 samples and used them to fit the discriminative DL model across 150 epochs. In the case of the power flow simulations, the convergence was not achieved, but the power flow was simulated over 100 iterations. We used 80 % of the available measurements for training and the remaining 20 % for validation.

Again, we chose the following four different combinations of multiples of the dimension of the partial measurement vector for the number of weights in each layer:

$$\{(5,5),(10,5),(15,10),(20,10)\} \tag{12}$$

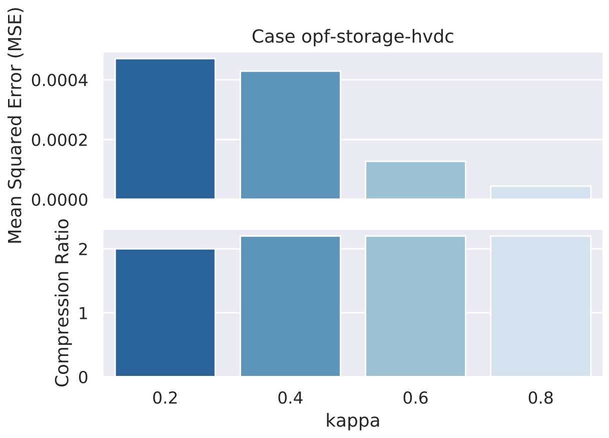


(a) Validation MSE and Compression Ratio as a function of parameter κ. The model architecture was fixed to $(10, 5)$.

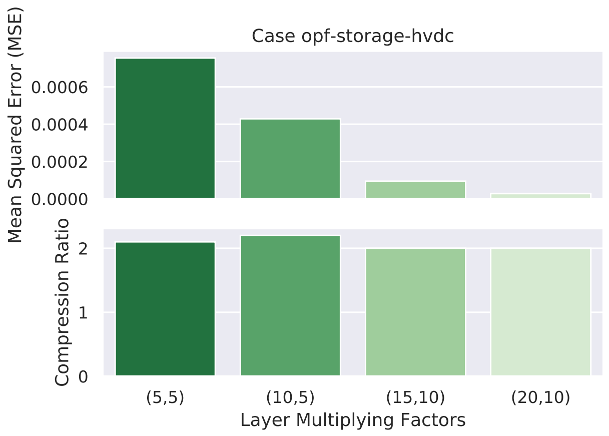


(b) Validation MSE and Compression Ratio as a function of Network Architecture (Multiplying Factors for Hidden Layers units)

Fig. 6: DL Modeling and Compression performance results for PyPSA Test Case 3.



(a) Validation MSE and Compression Ratio as a function of parameter κ. The model architecture was fixed to $(10, 5)$.



(b) Validation MSE and Compression Ratio as a function of Network Architecture (Multiplying Factors for Hidden Layers units)

Fig. 7: DL Modeling and Compression performance results for PyPSA Test Case opf-storage-hvdc.

We evaluated both the performance of the DL-base state estimation and the effectiveness of the Bayesian Compression method on the layers. For $\kappa \in \{0.2, 0.4, 0.6, 0.8\}$, we computed the final MSE on the validation set, and the compression factor, which is the ratio by which the model could be reduced, in terms of number of parameters, without having its performance affected. The results are shown in Figure 5. It can be seen that the compression ratio stays above 2 for a wide range of network architectures and observable nodes' ratio.

Regarding the MSE performance in the validation set, the network showed increased approximation capability as the number observed nodes are raised. However, regarding the network size, the most complex architecture ($(20, 10)$) showed a degradation in performance, possibly due to overfitting.

D. IEEE Test Case 57

We repeated the same steps as in Section III-C. Here, the model was trained for only 50 epochs, to prevent overfitting. The sensitivity analyses of the compression ratio and the DL model performance as a function of the model architecture and the partial measurements' percentage are shown in Figure 4. Regarding the compression ratio as a function of network architecture, it showed slightly better performance for the $(5, 5)$ and $(10, 5)$, perhaps as a result of overfitting of the larger models. Regarding the effect of number of observed nodes, the MSE performance showed an overall trend to improve as the number observed nodes were raised, although the κ with best result was 0.4. Regarding compression ratio, it showed some slight increase as the number of observed nodes rose, while also staying consistently above 2.

It was possible to observe that the Bayesian Compression method performed well across several network scales, but its performance is intrinsically dependent on the effectiveness of Power Flow optimization routines, which showed degraded performance for the larger 14s and 57 networks, not converging after the limit number of 100 iterations. Nevertheless, even with the power flow output variables which were not fully converged, the compression ratio stayed equal to or larger than

2.

IV. Conclusion

The Deep Learning (DL)-based approach to state estimation in unobservable Electrical Distribution Networks has recently been introduced, with promising results. However, the architecture choice and model training of deep neural networks is recognizably cumbersome and inefficient, possibly resulting in oversized models, with computationally inefficient layers. In the present work, we apply a technique know as "Bayesian Compression" to the DL modeling of state estimation in power networks, and show its efficacy in producing parsimonious but effective representations of the network underlying dynamics. The method is validated through experimental results in four test networks, including two IEEE standardized Test Case Networks.

References

[1] K. R. Mestav, J. Luengo-Rozas, and L. Tong, "Bayesian state estimation for unobservable distribution systems via deep learning," *CoRR*, vol. abs/1811.02756, 2018.

[2] K. R. Mestav and L. Tong, "Learning the unobservable: High-resolution state estimation via deep learning," in *57th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2019, Monticello, IL, USA, September 24-27, 2019.* IEEE, 2019, pp. 171–176.

[3] B. C. Csáji, "Approximation with artificial neural networks," Master's thesis, Faculty of Sciences, Eotvos Loránd University, Budapest, 2001.

[4] C. Louizos, K. Ullrich, and M. Welling, "Bayesian compression for deep learning," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 2017, pp. 3288–3298.

[5] "Solar home electricity data," 2014. [Online]. Available: https://www.ausgrid.com.au/Industry/Our-Research/Data-to-share/Solar-home-electricity-data

[6] G. Schwarz, "Estimating the dimension of a model," *Ann. Statist.*, vol. 6, no. 2, pp. 461–464, 03 1978. [Online]. Available: https://doi.org/10.1214/aos/1176344136

[7] T. Brown, J. Hörsch, and D. Schlachtberger, "PyPSA: Python for Power System Analysis," *Journal of Open Research Software*, vol. 6, no. 4, 2018. [Online]. Available: https://doi.org/10.5334/jors.188