POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

EXECUTIVE SUMMARY OF THE THESIS

Analysis of machine learning models to approximate the optimal power flow problem

TESI MAGISTRALE IN ELECTRICAL ENGINEERING – Sustainable Smart Grids for Energy Transition

AUTHOR: Nicolas Pietri

ADVISOR: Samuele Grillo

ACADEMIC YEAR: 2022-2023

1. Introduction

The energetic transition that we face involves to increase the use of the electrical network and so the use of tools which manage the power system as optimal power flow (OPF). OPF problem is a nonlinear, non-convex optimization problem that optimizes the operation of an electric power system by minimizing or maximizing an objective function within power flow equations constraints and operational limits. This problem is complex and computationally challenging. We investigated the use of machine learning techniques to solve it and speed up the process. We first investigated the OPF problem. Secondly, we defined artificial neural network (ANN). Third we made a review of the use of machine learning and more specifically deep learning to solve the OPF problem. Fourth we designed an ANN to solve the OPF problem.

2. Optimal power flow

The OPF problem was first presented in the 1960s by Carpentier [1] however it exists a wide range of applications, challenges, and requirements for OPF. The initial problem can be stated as

$$\min f(x) \qquad (1)$$

Where x are the different variables of the power system that we want to include in the problem as active/reactive power, voltage magnitude/angle… and f(x) is the objective function to minimize.
The problem is also subject to constraints:

$$g(x) = 0 \qquad (2)$$

$$h(x) \leq 0 \qquad (3)$$

where g(x) are the equalities constraints and h(x) are the inequalities constraints and x are the variables of the power system. x, f(x), g(x), h(x) vary respectively according to the states variables we want to include, the objective function we want to minimize, the equalities and inequalities we want to fulfill. f(x) can be the active generation cost [3], the non-supplied demand [4], the load curtailment costs [5], the active power losses [5], the tap changers and capacitor units [6]. The equalities g(x), and inequalities h(x) are related to the power flow equations and the operational limits. It exists an extensive number of possibilities to define x, f(x), g(x), h(x). This point explains that we can derive different types of OPFs. The solutions we decide to focus on are the AC OPF which respects the original power flow equations

and the AC network model, and DC OPF which is a linearized version of AC OPF. To solve the problem, there are several solving algorithms. The solutions obtained after solving the problem can be global, local or sub-optimal.

3. Artificial neural networks

As its name suggests, artificial neural networks are based on neural network, the output of a neuron can be computed with a forward propagation as

$$y = f\left(\sum_{i=0}^{n} x_i * w_i + b\right) \quad (4)$$

where y is the output, x a vector of inputs and w the weights associated with b a bias and f an activation function. The principle of neural network is the addition of layers connected to each other's successively with several neurons in each layer and this is the increase of the number of layers and neurons which allows to solve higher complex problems such as OPF. According to the input x the neural network computes the output y, this output is then compared to the expected output and the weight and bias are then updated with backward propagation:

$$w^{n+1} = w^n - \eta \frac{\partial J(w)}{\partial w} \quad (5)$$

$$b^{n+1} = b^n - \eta \frac{\partial J(w)}{\partial w} \quad (6)$$

With J(w) the loss error between the output predicted and the output expected, b the bias, w the weight, n the corresponding batch, η the learning rate. We can mention different activation function f, and loss error function J which implies a large scope of applications and ANN models.

4. State of the art of machine learning applied to the solving of optimal power flow

There are different kinds of use of machine learning to solve the optimal power flow problem. The most used method in the literature is the direct mapping of OPF variables to predict OPF solutions [7], [8], [9], [10]. Samples are generated for a set of OPF's input data, the OPF solution corresponding to each sample point is predicted. A learner is trained with the input data then obtains the OPF results (voltages, line flows, and power generation) as outputs. We chose this method as solution to solve the OPF problem however there exist other methods such as the predicting active constraints [11], [12], [13], [14], [15], [16]. It employs the sets of active constraints at optimality as the mapping output rather than the direct mapping of inputs to the optimal solution. Machine learning is also used to predict warm start points for the traditional solver [16], [29],[8] or used to solve stability constrained OPF [26],[27],[28]. We could also mention the use of machine learning, to learn the control policy for OPF [18], [19], [20], [21], [22], [23], [24],[25] or the mapping of binary decision variable [17].

As we are focusing on the mapping of OPF variables we can highlight key points of this process. The goal as mentioned is to predict directly as output of the ANN. The state variables of the system as the active power and voltage magnitude of the generator according to a given input, generally the active and reactive power of the loads are defined for AC OPF [30], [31]. For the DC OPF the inputs will be the active power of the load and the outputs will be the active power of the generators [36]. It has to be noticed that according to the method, other variables could be predicted such as voltage angle and magnitude of the buses or lagrangian multipliers for instance [33]. To generate data, we start from a given load set point and we vary the load from a given range according to a probability density function. Such as the uniform distribution preferred compared to the normal law, as our neural network has to perform well for the whole range of load variation and not only around the nominal load set point [34]. We use samples between [(1−y)∗Pd ; (1+y)∗Pd] with Pd the initial load active power set point and y following the uniform law:

$$y = \begin{cases} 1/(b-a) & for\ a \le x \le b \\ 0 & for\ x < a\ or\ x > b \end{cases} \quad (7)$$

With [a ; b] the interval where the probability density function is defined. The data is then normalized with standard core normalization or min-max feature. The latter approach is preferred as it allows to constrain the output by the additional use of a constrained output activation function as sigmoid function. The data is then process in the ANN. A wide range of ANN architecture are employed to solve OPF. It can be:

-A unique hidden layer feedforward neural network [39]

- A deep neural network constrained (as we used in our work) [36]

-A convolutional neural network [35]

-A graph neural network [32]

The benefit of deep neural network is the depth of the ANN, which allows a more optimal and accurate solution. It has to be noticed that dedicated activation function and loss functions are used in the literature in order to constrain the solution obtained, as, as mentioned before, OPF problem is a constrained optimization problem. As an example, we can mention that in some studies the output is constrained for the last layer with a sigmoid activation function [2], or also the use of penalty or barrier function which is added in the computation of the loss to take into account the constraints of the OPF problem while the model is training [2]. Lastly, post-processing step is done and metrics related to the performance of the ANN are computed. Post processing can include:

-To compute remaining variables of the problem as the loading of the line, lagrangian variables for instance [38]

-To ensure if the solution obtained is compliant with the constraints by using power flow [40]

-To use outputs obtained by the ANN as a warm start if the solution obtained is not compliant [31]

-To derive worst-case scenario according to the solutions obtained [37]

As different models have to be compared, metrics related to the performance of the neural network are computed. The first type of metric is related to the feasibility: it evaluates if the solution fulfills the constraints related to the OPF problem. The second type of metric is related to optimality: it evaluates if the solution is optimal according to the objective function we have to minimize/maximize (cost for instance) . The third type of metric is the accuracy of the solution obtained compared to the traditional solver and the last type of metric is related to the computational speed of the ANN to solve the OPF problem.

5. Deep neural network solution developed to solve optimal power flow

We built a deep neural network model to map the load input with the active power output of the generator. This work is inspired by the study [2] that we used as a reference to build our method.

There are three main parts: initialization, training, and test.

The initialization step,

We first develop the dataset by using uniform distribution, [-10%;+10%] and [-60%;+60%] variation of the active power of the load to generate samples of the input data and we use a DC OPF solver to compute the associated output which is the active power output of the generators. Input and output data are normalized using standard score normalization then secondly min-max feature scaling is applied to the output data also. We also apply pre-processing to compute elements related to the power system as the maximum active power of the lines

$$Pmax_{ij} = Imax_{ij} Vnom\sqrt{3} \quad (8)$$

B and A matrices are also computed, B matrix is used to recover the angle of the buses and A matrix to compute the DC power flow. A, is a matrix of size Nline × Nbus, where Nline is the number of lines in the power system and Nbus the number of buses in the power system. Each row in A corresponds to a line so a bus pair and each column corresponds to a bus. Thus, the elements, a_i and a_j are the corresponding entries of the matrix A defined as:

$$a_i = \frac{1}{Pmax_{ij}.x_{ij}}, a_j = -\frac{1}{Pmax_{ij}.x_{ij}} \quad (9)$$

where xij is the reactance of the line between bus i and bus j and Pmaxij defined above.

B matrix is an $(N_{bus}-1) \times (N_{bus}-1)$ admittance matrix derived from the admittance matrix Y by assuming that all voltages are 1pu, by neglecting shunt parameters of lines and transformers as well as resistances of lines and transformers. We can write the element of the B matrix as:

$$B_{ij} = \begin{cases} -y_{ij} \; for \; i \neq j \\ \sum_{j=1, j\neq i}^{N_{bus}} y_{ij} \;\; for \; i = j \end{cases} \quad (10)$$

where yij is the admittance of the branch yij with yij > 0 if there is a branch connected between node i and node j, and yij =0 if no branch is connected between bus i and bus j.

The training step,

This step is dedicated to how we build the deep neural network (DNN) and how we trained it. The DNN is made of one input layer corresponding to the size of the input dataset, there are 20 loads in the electrical network we use as example [41] so we will need 20 neurons as inputs to map the 20 load active power. Then two hidden layers of successively 64 and 32 neurons each with ReLU activation function, and one output layer of 6 neurons with sigmoid activation function to

constrain the output so that the active power of the generator is according to their operational limit. During the process we recover the voltage angle of the buses

$$\theta = B^{-1}P \quad (11)$$

where P is the active power injection at each bus. We express the power flow constraints of the DC OPF problem with:

$$-1 \leq A\theta \leq 1 \quad (12)$$

In order to take into account this constraint, we will use it as a penalty term in the loss error function:

$$Lerror = \frac{1}{Ntrain}\sum_{k=1}^{Ntrain} w1.Lpen_k + w2.MSE_k \quad (13)$$

$$MSE = \frac{1}{Ngen}\sum_{i=1}^{Ngen}(Ytrue_i - Ypredicted_i)^2 \quad (14)$$

$$Lpen = \frac{1}{Nline}\sum_{j}^{Nline} penalty(A\theta) \quad (15)$$

$$\text{penalty}(x) = x^2 - 1 \quad (16)$$

$$penalty(x) = \begin{cases} pf(x^2-1)\ for\ (x^2-1) > 0 \\ \frac{1}{pf}(x^2-1)\ for\ (x^2-1) < 0 \end{cases} \quad (17)$$

Where pf,w1,w2 are constants to be defined, Ngen the number of generator, Ntrain the number of training samples, $Ypredicted$ the predicted output of our deep neural network (DNN) model and $Ytrue$ the expected output of the traditional OPF solver. We defined and tried 2 penalty function (16)(17), in order to find the most efficient one to constrain effectively the solution obtained.

The test step,

In this step the goal is to test and evaluate the DNN that we trained. With a new dataset of inputs/outputs, by using our DNN and the active power of the loads we predict the active power of the generators and compare it to the expected active power of the generators. The metrics to compare these results are:

-The average time it takes to the neural network to compute the solution

-The mean square error between the predicted output and the expected output

By running DC power flow with the denormalized we then derive 2 other metrics related to the feasibility of our solution

-The percentage of network which are not valid

-The average error related to a load flow constraint violation:

$$Ierror = \frac{1}{Ntest}\sum_{l}^{Ntest}\frac{1}{Nline} \times \sum_{k}^{Nline}\frac{\left|min\left(max\_loading_{k,l} - Iloading_{k,l}, 0\right)\right|}{100} \quad (18)$$

Where $Ntest$ is the number of samples tested, Nline the number of lines in the network, $max\_loading_{k,l}$ the maximum loading of the line k for the sample l in percentage, $Iloading_{k,l}$ the loading of the line k for the sample l in percentage. The first results we report have been obtained with 10000 samples with [-10% ; +10%] variation of the load. We define 2 cases, each one describes the results obtained after the training of our DNN according to the error function used which is different for each case: Case 1, the loss function used to train our DNN is the mean square error, it corresponds to the equations (13), (14) with w1=1 and w2=0 as parameters. Case 2, the loss function used to train our DNN is the mean square error in addition to the penalty function, it corresponds to the equations (13), (14), (15), (16) with w1=1 and w2=0.001

	MSE	$Average\ time$ [s]	$Ierror$	Percentage of network which are not valid
Case 1	5.97890e-06	0.000578	0	0
Case 2	5.97890e-06	0.000703	0	0

Table 1: Results 1 of the DC neural network

As results we can conclude that our DNN model increases the solving speed of the OPF problem by approximatively 100 times the traditional OPF problem. We also see that with or without penalty function there are no changes. It means that the penalty function used has no influence and so that our neural network does not learn the limits/constraints related to optimal power flow and related to the active power limit of the lines. As we vary the load only between [-10%;+10%], this peculiar case leads to the fact that the results are always far away from the maximum limits of our network. So, we solve only a small range of the problem. In order to test our neural network correctly, we generate samples with active power of the load varying with a range [-60%;+60%] from the initial setpoints which is also a closer approximation of the real load range variation. We keep the same number of samples (10,000) and other parameters. We define 3 cases, each one describes the results obtained after the training of our DNN according to the error function used which is different for each case :

Case 1 The loss function is defined as the mean square error only, it corresponds to the equations (13), (14) with w1=1 and w2=0)
Case 2 The loss function is defined as the sum of the mean square error and the penalty function, it corresponds to the equations (13), (14), (15), (16) with w1=1 and w2=0.001
Case 3 The loss function is defined as the sum of the mean square error and the penalty function, it corresponds to the equations (13), (14), (15), (17) with w1=1, w2=1, pf=10000

	MSE	$Average\ time$ [s]	$Ierror$	Percentage of network which are not valid
Case 1	0.00055	0.00041	0.06146	15,75%
Case 2	0.00055	0.00035	0.06146	15,75%
Case 3	0.00061	0.000776	0.07568	13,1%

Table 2: Results 2 of the DC neural network

We see that the penalty function used in case 2 is not relevant to reduce the violation of constraints. In case 3 we reduce the percentage of network which are not valid however when a network is not valid the error obtained is larger as Ierror is worst in case 3 than case 1 and 2. It means that we will have fewer violations of the constraints in term of number but larger in term of magnitude. The case3 shows that we are probably underfitting, as we increase the percentage of variation of the load. It means that the range of the solution that the neural network has to learn is also wider thus needing more data to be trained. We train 58,360 samples with [-60%;60%] uniform variation, then we define 3 cases. Each one describes the results obtained after the training of our DNN according to the error function used which is different for each case:
Case 1 The loss function is defined as the mean square error only, it corresponds to the equations (13), (14) with w1=1 and w2=0
Case 2 The loss function defined as the sum of the mean square error and the penalty function, it corresponds to the equations (13), (14), (15), (16) with w1=1 and w2=0.001
Case 3 The loss function is defined as the sum of the mean square error and the penalty function, it corresponds to the equations (13), (14), (15), (17) with w1=1, w2=1, pf=10000

	MSE	$Average\ time$ [s]	$Ierror$	Percentage of network which are not valid
Case1	0.00011	0.00047	0.18806	11.99%
Case 2	0.00011	0.00069	0.18806	11.99%
Case 3	0.00018	0.00259	0.16120	10.49%

Table 3: Results 3 of the DC neural network

We see that the penalty term we introduce for case 3 leads to a better feasibility as it reduces the magnitude of the violation of the constraint (Ierror) but it also reduces the number of violations of the constraints. However, we see that the difference is not so significant between case 1 and case 3 and also adding a penalty term as we already said will decrease the accuracy (MSE) of the solution obtained. Note that we are also probably still underfitting otherwise it would become too computationally challenging to increase the number of samples.

Finally, we do a recovery of the results. If the post-processing leads to infeasible solution, we recompute new solutions with traditional OPF solver. The best would be to use the outputs predicted as a warm start point in order to make the algorithm converge faster. At the end of the process, we will get solutions which ensure to solve the DC OPF problem without violation of the constraints.

In this part we also defined the theory related to an AC OPF deep learning method. We keep the [-60%;+60%] variation with uniform distribution but this time using a traditional AC OPF solver. As input of the DNN model we will have active/reactive power of the loads and as outputs the voltage magnitude/angle of the buses and active/reactive power of the generators. We would define the architecture of the DNN with 1 common input layer and 4×2 hidden layers in parallels. It means 2 hidden layers for each variable, where we will use ReLU activation function. At the end we will use 1 common output layer which would be with a sigmoid activation function to constraints the output. To deal with the power flow constraints of the cable we could use the formula:
$S_{ij} = (v_i^2 - v_j v_i(\ cos(\theta_{ij}) + jsin(\theta_{ij}))\ (\ g_{ij} - jb_{ij})$

(19)

Where v_i , v_j are the voltage magnitudes of the buses i and j where the line is connected, θ_{ij} the voltage angle difference between bus i and j, g_{ij} and b_{ij} the equivalent conductance and susceptance of the line. S_{ij} the apparent power flow in the line. We could keep the same loss error functions (13), (14), (15), (17) with w1=1, w2=1, pf=10000.

To test this AC OPF DNN model, we would use the same metrics and process compared to the DC OPF DNN model except the fact that we must check the feasibility of the model by using AC power flow and not DC power flow.

6. Conclusions

The use of deep learning to solve OPF problem is still a challenging problem because of the constraints of power flow to fulfil and the huge number of data it requires to train the model to learn these constraints and the optimal solutions. Nevertheless, from the first trial we achieved that artificial neural network could effectively approximate solutions of the OPF problems. During our work we developed a general method to build a deep neural network according to a given electrical network in order to solve DC OPF problem. There are many possible areas for improvement of the study:

- Use of a warm start solver at the end of our method to recover the solutions when the predicted solution is infeasible
- Speed-up the training process and the sample generation in order to deal with a larger dataset
- Achieve the part of the AC neural network, as we just designed in theory the neural network related to this part so we need now to build it correctly and test it
- Find an effective barrier or penalty functions to constraints effectively our DNN model
- Try solutions different from the simply mapping of the variables and compared their effectiveness

7. Bibliography

[1] Carpentier J. Contribution à l'étude du dispatching économique. Bull de la Société Française des Electriciens 1962;3:431–47.

[2] Xiang Pan, Tianyu Zhao, Minghua Chen. DeepOPF: Deep Neural Network for DC Optimal Power Flow.

[3] Lin S-Y, Chen J-F. Distributed optimal power flow for smart grid transmission system with renewable energy sources. Energy 2013;56:184–92.

[4] Sousa T, Morais H, Vale Z, Castro R. A multi-objective optimization of the active and reactive resource scheduling at a distribution level in a smart grid context. Energy 2015;85:236–50.

[5] Bruno S, Lamonaca S, Rotondo G, Stecchi U, Scala M La. Unbalanced three-phase optimal power flow for smart grids. IEEE Trans Ind Electron 2011;58:4504–13.

[6] Paudyal S, Canizares CA, Bhattacharya K. Optimal operation of distribution feeders in smart grids. IEEE Trans Ind Electron 2011;58:4495–513.

[7] Y. Ng, S. Misra, L. A. Roald, and S. Backhaus, "Statistical learning for DC optimal power flow," IEEE Power Systems Computation Conference, 2018.

[8] Y. Sun, X. Fan, Q. Huang, X. Li, R. Huang, T. Yin, and G. Lin, "Local Feature Sufficiency Exploration for Predicting Security-Constrained Generation Dispatch in Multi-area Power Systems," 17th IEEE International Conference on Machine Learning and Applications, 2018, pp. 1283-1289.

[9] T. Navidi, S. Bhooshan, and A. Garg, "Predicting Solutions to the Optimal Power Flow Problem," Project Report, Stanford University, 2016

[10] R. Canyasse, G. Dalal, and S. Mannor, "Supervised learning for optimal power flow as a real-time proxy," IEEE Power & Energy Society Innovative Smart Grid Technologies Conference, 2017.

[11] S. Misra, L. Roald, and Y. Ng, "Learning for Constrained Optimization: Identifying Optimal Active Constraint Sets," arXiv preprint arXiv:1802.09639, 2018.

[12] D. Deka and S. Misra, "Learning for DC-OPF: Classifying active sets using neural nets," arXiv preprint arXiv:1902.05607, 2019.

[13] A. J. Ardakani and F. Bouffard, "Prediction of Umbrella Constraints," IEEE Power Systems Computation Conference, 2018.

[14] K. Baker and A. Bernstein, "Joint chance constraints reduction through learning in active distribution networks," IEEE Global Conference on Signal and Information Processing, 2018, pp. 922-926.

[15] K. Baker and A. Bernstein, "Joint Chance Constraints in AC Optimal Power Flow: Improving Bounds through Learning," IEEE Transactions on Smart Grid, 2019.

[16] A. S. Xavier, F. Qiu, and S. Ahmed, "Learning to Solve Large-Scale Security-Constrained Unit Commitment Problems," arXiv preprint arXiv:1902.01697, 2019.

[17] G. Dalal, E. Gilboa, S. Mannor, and L. Wehenkel, "Unit commitment using nearest neighbor as a short-term proxy," IEEE Power Systems Computation Conference, 2018.

[18] R. Dobbe, O. Sondermeijer, D. Fridovich-Keil, D. Arnold, D. Callaway, and C. Tomlin, "Data-driven decentralized optimal power flow," arXiv preprint arXiv:1806.06790, 2018.

[19] O. Sondermeijer, R. Dobbe, D. Arnold, C. Tomlin, and T. Keviczky, "Regression-based inverter control for decentralized optimal power flow and voltage regulation," arXiv preprint arXiv:1902.08594, 2019.

[20] A. Garg, M. Jalali, V. Kekatos, and N. Gatsis, "Kernel-based learning for smart inverter control," IEEE Global Conference on Signal and Information Processing, 2018, pp. 875-879.

[21] F. Bellizio, S. Karagiannopoulos, P. Aristidou, and G. Hug, "Optimized local control for active distribution grids using machine learning techniques," IEEE Power & Energy Society General Meeting, 2018.

[22] S. Karagiannopoulos, P. Aristidou, and G. Hug, "Data-driven Local Control Design for Active Distribution Grids using off-line Optimal Power Flow and Machine Learning Techniques," IEEE Transactions on Smart Grid, 2019.

[23] T. Summers, J. Warrington, M. Morari, and J. Lygeros, "Stochastic optimal power flow based on convex approximations of chance constraints," IEEE Power Systems Computation Conference, 2014.
[24] Y. Guo, K. Baker, E. Dall'Anese, Z. Hu, and T. H. Summers, "DataBased Distributionally Robust Stochastic Optimal Power Flow—Part II: Case Studies," IEEE Transactions on Power Systems, vol. 34, no. 2, pp. 1493-1503, 2018.
[25] Y. Guo, K. Baker, E. Dall'Anese, Z. Hu, and T. H. Summers, "Databased distributionally robust stochastic optimal power flow—Part I: Methodologies," IEEE Transactions on Power Systems, vol. 34, no. 2, pp. 1483-1492, 2018.
[26] F. Thams, L. Halilbasic, P. Pinson, S. Chatzivasileiadis, and R. Eriksson, "Data-driven security-constrained opf," in X Bulk Power Systems Dynamics and Control Symposium, 2017.
[27] L. Halilbašiü, F. Thams, A. Venzke, S. Chatzivasileiadis, and P. Pinson, "Data-driven security-constrained AC-OPF for operations and markets," IEEE Power Systems Computation Conference, 2018.
[28] R. T. A. King, X. Tu, L.-A. Dessaint, and I. Kamwa, "Multi-contingency transient stability-constrained optimal power flow using multilayer feedforward neural networks," IEEE Canadian Conference on Electrical and Computer Engineering, 2016.
[29] K. Baker, "Learning Warm-Start Points for AC Optimal Power Flow," arXiv preprint arXiv:1905.08860, 2019.
[30] Machine Learning for AC Optimal Power Flow, Neel Guha, Zhecheng Wang, Matt Wytock, Arun Majumdar
[31] Learning Optimal Solutions for Extremely Fast AC Optimal Power Flow, Ahmed S. Zamzam and Kyri Baker
[32] OPTIMAL POWER FLOW USING GRAPH NEURAL NETWORKS, Damian Owerko, Fernando Gama, and Alejandro Ribeiro
[33] Physics-Informed Neural Networks for AC Optimal Power Flow Rahul Nellikkath, Spyros Chatzivasileiadis
[34] Learning to Solve the AC-OPF Using Sensitivity-Informed Deep Neural Networks, Manish K. Singh, Vassilis Kekatos, Georgios B. Giannakis
[35] A CNN Approach for Optimal Power Flow Problem for Distribution Network, Yujing Jia, Xiaoqing Bai
[36] DeepOPF: Deep Neural Network for DC Optimal Power Flow, Xiang Pan, Tianyu Zhao, Minghua Chen
[37] Physics-Informed Neural Networks for Minimising Worst-Case Violations in DC Optimal Power Flow, Rahul Nellikkath, Spyros Chatzivasileiadis
[38] DeepOPF-NGT: Fast No Ground Truth Deep Learning-Based Approach for AC-OPF Problems, Wanjun Huang, Minghua Chen
[39] Learning Optimal Power Flow: Worst-Case Guarantees for Neural Networks, Andreas Venzke, Guannan Qu, Steven Low, Spyros Chatzivasileiadis
[40] Compact Optimization Learning for AC Optimal Power Flow, Seonho Park, Wenbo Chen, Terrence W.K. Mak, Pascal Van Hentenryck
[41]https://labs.ece.uw.edu/pstca/dyn30/pg_tcadyn30.htm