Electrical Power and Energy Systems 130 (2021) 106908

Contents lists available at ScienceDirect

International Journal of Electrical Power and Energy Systems

journal homepage: www.elsevier.com/locate/ijepes

A learning-augmented approach for AC optimal power flow

Jubeyer Rahman, Cong Feng, Jie Zhang *

The University of Texas at Dallas, Richardson, TX 75080, USA

ARTICLE INFO

Keywords:
AC OPF
Random forest
Decision tree
Extreme learning machine
Optimality

ABSTRACT

Due to the high nonlinearity of AC optimal power flow (OPF), numerous efforts have been made in recent decades to find efficient methods. Machine learning (ML) has proven to significantly reduce the computational costs in many real-world problems. Thus, this paper develops a learning-augmented method for solving AC OPF, which integrates both power network equations and ML to yield near-optimal solutions. More specifically, ML models are developed to first predict bus voltage magnitudes and angles. Then, physics-based network equations are employed to calculate the power injection at different buses. Three ML algorithms, i.e., random forest, multi-target decision tree, and extreme learning machine, are explored and compared. To evaluate the efficiency of the proposed learning-augmented AC OPF solver, the MATPOWER Interior Point Solver is adopted as a baseline. Case studies on both 500-bus and 4918-bus test networks show that the proposed learning-augmented method has reduced the computational time by 15–100 times depending on the network size with a minimal loss in optimality.

1. Introduction

Under present deregulated electricity market structures, system operators require precise dispatch and market decisions to better utilize available resources and preserve interests of all market stakeholders [1]. Independent System Operators (ISOs) execute optimal power flow (OPF) algorithms to obtain their dispatch decisions frequently (e.g., California ISO runs OPF every 5-min [2]). Due to the uncertain and stochastic nature of renewable resources, OPF decision parameters need to be determined promptly to avoid their curtailments. It has been reported that alone in the U.S., a 5% increase in market efficiency with better OPF solvers, can save approximately 6 billion dollars per year [3]. However, AC OPF is still a challenging problem, due to its nonlinearity, especially for large networks. Many approaches to solve the full AC-OPF have been proposed in recent decades, including linear relaxation [4,5], convex relaxation [6], and heuristic algorithms [7–9].

1.1. Review of machine learning approaches to OPF

Machine learning (ML) has demonstrated to significantly improve the computational efficiency of many complex problems, including OPF [10–13]. The utilization of ML approaches to solve OPF can be broadly divided into two categories: (1) end-to-end approaches that are purely based on ML, and (2) hybrid approaches of ML and physics-based solvers. A number of end-to-end ML methods have been developed in the literature to solve OPF. For example, a collection of supervised learning algorithms, including Gaussian naive Bayes, logistic regression, decision tree, random forest, extra-tree, and neural network, were adopted to only solve the objective function value of OPF [14]. In Ref. [15], system real power load and costs of generation were used to predict generators' real power output and the voltage of P-V buses using several ML models. The prediction of active OPF constraints was studied in [10–12,16,17] to improve the computational efficiency of traditional physics-based solvers. An end-to-end generator setting prediction was performed in [12] through a constraints violation penalty-incorporated neural network approach. A deep learning technique was exploited in [18] to solve the real power generation in DC OPF, where post-processing was performed to ensure that the generation values were within bounds. To reduce the feasibility gap, the loss function of a deep neural network was customized to the lagrangian of the AC OPF problem in [19]. A distributed reinforcement learning-based approach was proposed in [20], where real and reactive power losses were minimized through a feedback scheme.

Hybrid approaches were also attempted in the literature to reduce the computational cost, focusing primarily on yielding better warm-start points for physics-based solvers. For example, a neural network model was used in Ref. [21] to represent the system security boundary and generate new constraints in the form of a differentiable mapping

* Corresponding author.
E-mail address: jiezhang@utdallas.edu (J. Zhang).

https://doi.org/10.1016/j.ijepes.2021.106908
Received 28 August 2020; Received in revised form 21 December 2020; Accepted 26 January 2021
Available online 21 March 2021
0142-0615/© 2021 Elsevier Ltd. All rights reserved.

function, which was then solved by a physics-based OPF solver. A multi-input multi-output random forest regression method was used in [22] to predict the real power set-point of generators and bus voltage magnitudes, which were then passed to a physics-based OPF solver. An artificial neural network model was used in [23] to only solve generator voltage and real power outputs, which were then passed to a physics-based solver to solve the reactive power.

1.2. Research motivation and objective

The main drawback of using ML to obtain direct end-to-end solutions is that there might exist a large number of violations with those physical law-related constraints. It is noted in the literature that a majority of end-to-end ML-based OPF methods only solve real power generation of generators and bus voltage magnitudes, which may lead to infeasible solutions. This is due to the fact that both the voltage magnitude and angle determine the amount of power flow in branches, and the power losses of lines are also dependent on those voltage parameters. Thus the real power generation obtained from end-to-end ML models may not be useful, since they may contradict the injection values calculated from predicted voltage parameters based on network equations. While hybrid approaches can ensure the convergence, physics-based solvers are still required. Thus the computational efficiency improvement of hybrid approaches is still limited.

To address the challenges in both ML-based end-to-end and hybrid approaches, this paper develops a learning-augmented approach to solve AC OPF, which does not rely on any physics-based OPF solvers. To address the issues of physical law related constraints violation, the proposed model solves the voltage magnitudes and angles instead of real or reactive power generation. The generated voltage magnitude and angles won't contradict each other as per the physical laws, which guarantees the feasibility of the OPF solutions. Once the voltage magnitudes and angles are obtained, physics-based power flow equations are applied to calculate other operating variables, such as real power generation, shunt susceptances, etc. By integrating the physical laws with the computationally efficient ML-based voltage parameter prediction scheme, the proposed approach is expected to (i) reduce the input–output mapping dimension for the ML algorithm and thus reduce the computational expense; (ii) ensure the solutions' compliance with physical laws without any physics-based OPF solvers.

In addition, the proposed method doesn't rely on conventional OPF solvers, which can significantly reduce the computational time, especially for large-size networks, e.g., the U.S. eastern-interconnection (62,000 bus and 80,000 branch system [24]), or scenarios where a large number of OPF cases need to be solved within a short time frame, e.g., contingency analysis, real-time economic dispatch with stochastic renewable generation, etc. AC OPF is usually considered as a non-deterministic polynomial (NP)-hard problem, and improving its computational efficiency has always been an appealing task [19,25–28]. Due to the NP-hard nature of AC OPF, the solution cannot be guaranteed to be obtained in polynomial time [25], which forces the system operators to resort to a linearized version of AC OPF (i.e., DC OPF) for dispatch decisions and market clearing signals [14,27]. To address this challenge, the proposed method could be an alternative solution for real-time dispatch decision and spot market clearing.

The rest of the paper is organized as follows. In Section 2, the proposed learning-augmented approach for AC OPF is described along with the ML model development. In Section 3, the experimental setup is described. Section 4 presents detailed results and discussion of case studies. Section 5 concludes the paper and briefly discusses potential future work.

2. Methodology

2.1. The OPF problem

The OPF problem deals with determining the optimal dispatch of generators to minimize the cost of generation while satisfying the engineering and physical constraints. In this paper, the OPF formulation of the 'Grid Optimization (GO)' competition [29] is adopted and modified. The 'GO' competition formulation contains controllable shunt related security constraints, in addition to commonly considered constraints in the literature. Thus, the AC OPF problem in this paper is formulated as follows.

$$\min_{v,\theta,p,q,b^{cs}} \sum_{g=1}^{G} c_g$$

s.t.

$$\begin{aligned}
&\underline{v}_i \leqslant v_i \leqslant \overline{v}_i,\ \forall i \in \mathcal{N}\\
&\underline{\theta}_i \leqslant \theta_i \leqslant \overline{\theta}_i,\ \forall i \in \mathcal{N}\\
&\underline{p}_g \leqslant p_g \leqslant \overline{p}_g,\ \forall g \in \mathcal{G}\\
&\underline{q}_g \leqslant q_g \leqslant \overline{q}_g,\ \forall g \in \mathcal{G}\\
&\underline{b}_k^{cs} \leqslant b_k^{cs} \leqslant \overline{b}_k^{cs},\ \forall k \in \mathcal{B}\\
&p_{inj}^i - p_d^i = \sum_{(i,j)\in\mathcal{E}} p_{i,j}\left(v,\theta\right),\ \forall i \in N, \forall\left(i,j\right) \in \mathcal{E}\\
&q_{inj}^i - q_d^i = \sum_{(i,j)\in\mathcal{E}} q_{i,j}\left(v,\theta\right),\ \forall i \in N, \forall\left(i,j\right) \in \mathcal{E}\\
&\sqrt{p_{i,j}^2 + q_{i,j}^2} \leqslant \overline{s_{i,j}},\ \forall\left(i,j\right) \in \mathcal{E}
\end{aligned} \tag{1}$$

where $\mathcal{N}, \mathcal{E}, \mathcal{B}$, and $\mathcal{G}$ are the set of bus, branch (both line and transformer), switched shunt, and generator, respectively. The parameter c_g represents the generation cost of generator g, which is calculated from an individual piece-wise linear cost function. The parameters v_i and θ_i are the voltage magnitude and angle at bus i, respectively. Generator active and reactive power is denoted by p_g and q_g, respectively. The shunt susceptance is denoted by b^{cs} and limited by their MVAR limits. The parameters p_{inj}^i and q_{inj}^i stand for the real and reactive power injection at bus i, whereas p_d^i and q_d^i are the real and reactive power demand at that bus, respectively. The net real and reactive power injection $p_{i,j}$ and $q_{i,j}$ to a branch can be represented by the difference between injection and demand at the connected buses i and j. For a branch connecting bus i and j, the power flow $s_{i,j}$ is limited by its rating $\overline{s}_{i,j}$. A feasible solution ensures that these constraints are not violated. The constraints derived from physical laws like Ohm's law and Kirchhoff's law (e.g., equality), have zero tolerance for violation. The constraints that are related to engineering practice could be relaxed at times such as, during contingency events. In this work, for data generation purpose, the branch flow limit is changed to a soft-limit when the case is not solvable by treating the branch flow-limit as a hard-limit. This constraint relaxation allows for a larger phase variation in required cases. In a mathematical formulation, this constraint relaxation can be described as:

$$-\overline{s_{i,j}} - \lambda \leqslant s_{i,j} \leqslant \overline{s_{i,j}} + \lambda \tag{2}$$

where λ is a slack variable that allows a headroom for the solver to enhance its feasible solution space.

2.2. Learning-augmented OPF methodology

Figure 1 shows the overall framework of the proposed learning-augmented approach for solving AC OPF. With given load conditions

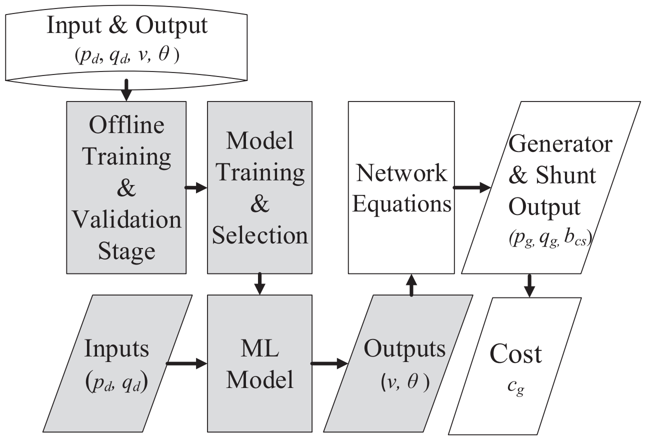

Fig. 1. The proposed learning-augmented AC OPF algorithm. The shaded region represents the ML domain.

(i.e., real and reactive power load), corresponding voltage magnitudes and angles are solved using a conventional OPF solver (i.e., The MATPOWER Interior Point Solver (MIPS) in this paper), which generates a large dataset for ML model training. To successfully train the machine learning model, a large training dataset (generated from offline optimization-based OPF solutions) are needed. The proposed method could be used in a number of scenarios when real-time OPF solutions are needed, such as (i) real-time dispatch decision under contingencies, disasters, or cyber attacks; (ii) potential future power networks with high penetrations of uncertain renewable generation. In addition, during normal conditions, the learning-augmented solution could also be used as a warm-start for the consecutive solution, since the system states vary little within a short time frame without major disruption [19]. Thus in the model training process, the training data (i.e., generated input load data and output OPF solution from the conventional MIPS solver) is necessary to make the proposed learning-augmented model be aware of the system's typical parameter and decision variable setting. The generated dataset is split into training, validation, and testing datasets. A set of state-of-the-art ML regression models are trained based on the training dataset and optimized using the validation dataset. The trained models are later used for making voltage magnitude and angle predictions. Since the inputs to the proposed learning-augmented model are bus loads and the outputs are voltage magnitudes and angles, this mapping operator is written as:

$$\Omega : P_d \times Q_d \longrightarrow \mathcal{V} \times \Theta \tag{3}$$

where P_d and Q_d are the sets for real and reactive bus load, respectively; $\mathcal{V}$ and Θ are the sets for bus voltage magnitude and angle, respectively. This mapping function Ω is approximated by an ML model.

After solving the voltage parameters, network equations are applied to determine the current injection I^{inj} at different buses, based on a bus admittance matrix Y that is usually calculated from given network parameters.

$$I^{inj} = Y \times v \tag{4}$$

Once current injections are known, the apparent power S is calculated from the product of bus voltage and current conjugate.

$$S = v \times I^{inj^*} \tag{5}$$

Finally, the complex bus power injection S_{inj} is calculated by subtracting the bus power demand S_d from the apparent power. This is inspired from the repetitive adjustment of system voltage parameters to produce a zero mismatch in complex bus power injection, which is typically used in physics-based power flow algorithms.

$$S_{inj} = S - S_d \tag{6}$$

The real part of the complex bus power injection at the generator bus is then fed into the piece-wise linear cost functions of individual generators to calculate the overall cost of production. The learning-augmented OPF solver, i.e., the integration of ML and network equations, ensures that the constraints related to physical laws are satisfied. Regarding the cost or objective function value, the learning-augmented model produces the OPF solution based on the trained model. Since the training is performed with the generated OPF solutions that minimize the cost, the learning-augmented solver is also expected to minimize the cost.

2.3. ML models

The developed ML models for solving AC OPF should have a multi–input–multi–output (MIMO) mapping capability, with a high computational speed and accuracy. Most of the existing ML algorithms don't have the MIMO handling capability and have to rely on ensemble techniques to accomplish MIMO tasks, especially for regression problems. However, the computational efficiency may get significantly affected by ensemble methods, and the performance may deteriorate through the ensembling process [30,31]. Tree-based and nearest neighbor algorithms have shown to perform well in multi-target mapping. However, nearest neighbor methods are notoriously slow [25] and thus unsuitable. Thus, two tree-based ML algorithms are adopted and compared to perform the learning task, which are Random Forest (RF) and multi-target Decision Tree (DT). An extremely fast single layer feedforward neural network (SLFN), called the Extreme Learning Machine (ELM), is also adopted for comparison, due to its automatic feature selection capability and fast training. Several deep learning algorithms, such as the deep feed-forward neural network and deep 1-dimensional convolutional neural network, were also tested due to their automatic feature selection capability. However, the accuracies of the tested deep learning algorithms are not satisfactory and their results are not reported here.

2.3.1. Random forest model

RF has a superior multi-input multi-output mapping capability thus perfectly suits the purpose of estimating multiple AC OPF variables. RF is a combination of bagging [32] and random feature subspacing [33], both of which are ensemble techniques trained with a particular prediction scheme [34]. There exist a collection of decision trees in an RF model to make their predictions separately. Then, these predictions are averaged to minimize the error. The trees are base learners and near-independent, which reduces the risk of biased decision or overfitting.

During the training of the RF model, bootstrap samples are drawn from the input dataset (i.e., the system load P_d and Q_d), and for each sample, an unpruned regression tree is grown. At every node, input variables are sampled randomly and splitting is performed in the variable space. The best split is ensured by minimizing the residual sum of squares (RSS), given by:

$$RSS = \sum_{l=1}^{L} \sum_{m \in R_l} \left(y_m - \hat{y}_{R_l}\right)^2 \tag{7}$$

where, $\hat{y}_{R_l}$ is the average of the observations in the l^{th} non-overlapping region, L is the total number of regions of the predictor space, and y_m is the predicted value. The training process continues until the number of grown trees is equal to a predefined number. The model outputs (i.e., v and θ) are generated by aggregating the prediction of all grown trees. Here, the average solution of all trees is considered as the final solution, which ensures zero random prediction errors of the trees (forest) and preserves the true relationship between predictors and responses. The pseudocode for RF-based OPF is described in Algorithm 1. Unlike other ML algorithms, cross-validation is not required in RF since an out-of-bag

(OOB) error estimation is used during the tree construction [32].

Algorithm 1. RF-based OPF Algorithm

```
Require: RF training and test dataset: $(X_{tr}, Y_{tr})$, $(X_{test}, Y_{test})$
  RF model parameter (No. of trees $B$)
Ensure: OPF variables are determined
1: RandomForest($(X_{tr}, Y_{tr}), B$)
2: Initialize $Q = \phi$
3: for e = 1 to B do
4:   $I_e \leftarrow$ a bootstrap sample from training set
5:   $q_e \leftarrow$ get the learned tree from the sample with a cutpoint to minimize Eq. (7)
6:   $Q \leftarrow Q \cup q_e$
7: end for
8: return $Q$
9: Predict with $Q$ on $X_{test}$ to get $v$ and $\theta$
10: Determine other OPF variables by using Eqs. ()()()(4)–(6)
```

2.3.2. *Multi-target decision tree*

Multi-target DT is capable of doing multi-target regression at one time by a single tree instead of building one for each target variable. The multi-target DT can exploit the benefits of correlation among input variables. Compared to a single target regression tree, multi-target DT is significantly smaller and requires less time to train. Unlike storing a single numeric value at each leaf node, each multi-target DT leaf stores a vector. The learning begins at the root node of a single DT with a set of training data, and the set is recursively partitioned into smaller subsets by a heuristic function [35]. Akin to the single-target tree, a heuristic function selects the input variable set at each internal node of a multi-target DT on the basis of intra-cluster variation $N\sum_{t=1}^{T} var[y_t]$, where T is the number of target variables, N is the number of samples, and $var[y_t]$ is the variance of the target variable y_t in the cluster [35]. A smaller value of intra-subset variance represents a higher accuracy in prediction. The partitioning process stops when a predefined stop criterion (i.e., the maximum tree depth in this paper [36]) is satisfied.

Algorithm 2. DT-based OPF Algorithm

```
Require
  DT training and test dataset: $(X_{tr}, Y_{tr})$, $(X_{test}, Y_{test})$
Ensure: OPF variables are determined
1: DT($(X_{tr}, Y_{tr})$)
2: Initialize $tree = \phi$
3: Initialize $k$ cluster set $C_k = (X_{tr,k}, Y_{tr,k})$
4: $C^* \leftarrow$ get the best cluster set by minimizing intra-cluster variance
5: $y^* \leftarrow$ get the average value of targets at leaf node
6: if $y^* = \phi$ then
7:   for each $C_k^* \in C^*$ do
8:     $tree \leftarrow$ DT($(X_{trk}, Y_{trk})$)
9:   end for
10:    return $node(y^*, tree)$
11: else
12:    return $leaf(y^*)$
13: end if
14: return $tree$
15: Predict with $tree$ on $X_{test}$ to get $v$ and $\theta$
16: Determine other OPF variables by using Eqs. ()()()(4)–(6)
```

Then, the target variables are calculated and averaged for each leaf node to obtain the final prediction. An averaging takes place at the leaf node to get the prediction for each target variable for the considered observation. The pseudocode for DT-based OPF is illustrated in Algorithm 2.

2.3.3. *Extreme learning machine*

Since both load and generation situations change rapidly in power systems, the learning speed is critically important when solving OPF, especially under extreme conditions such as contingencies. ELM has shown to be capable of reducing both the training and prediction time significantly, compared to classic ML algorithms. The low computational efficiency of classic ML algorithms is mainly attributed to: i) slow gradient-based learning algorithms, and ii) the hectic iterative tuning of model parameters to optimize their performance. ELM addresses the aforementioned challenges by randomly choosing input weights and analytically determining the output weights. The input weights and hidden layer biases are assigned at the start of the learning process. The hidden layer output matrix (H) stays unaltered if the number of hidden nodes is equal to the number of distinct input samples. If an exception occurs, the output weights are calculated by using the *Moore–Penrose generalized inverse* of matrix H instead of using H directly [37]. For a single target variable, an ELM with $\widetilde{N}$ hidden neurons is described as follows.

$$f_{\widetilde{N}}(x) = \sum_{i=1}^{\widetilde{N}} \beta_i h_i(x) = h(x)\beta \tag{8}$$

where x is the input, β is the output weight vector, and $h(x)$ is the hidden layer output vector. For a multi-target regression problem, $f_{\widetilde{N}}$ is an $M \times \widetilde{N}$ matrix where M is the number of target variables. The cost function E is minimized with N samples.

$$E = \sum_{j=1}^{N} \left(\sum_{i=1}^{\widetilde{N}} \beta_i g\left(\mathbf{w}_i \cdot \mathbf{x}_j + b_i\right) - \mathbf{y}_j \right)^2 = ||H\beta - \mathbf{y}||^2 \tag{9}$$

where $g(\cdot)$ is an activation function that is assumed to be infinitely differentiable. The parameter $\mathbf{w}_i$ is a weight vector connecting the input node and the i^{th} hidden nodes. The parameter b_i is a threshold of the i^{th} hidden node. $\mathbf{w}_i \cdot \mathbf{x}_j$ denotes the inner product between $\mathbf{w}_i$ and $\mathbf{x}_j$, $\mathbf{y}_j$ represents a target vector, and $\mathbf{y}$ represents a target matrix. The pseudocode for ELM-based OPF is illustrated in Algorithm 3.

Algorithm 3. ELM-based OPF Algorithm

```
Require
  ELM training and test dataset: $(X_{tr}, Y_{tr})$, $(X_{test}, Y_{test})$
ELM model parameters (No. of hidden neuron $\widetilde{N}$, activation function $g(x)$)
Ensure: OPF variables are determined
1: ELM($(X_{tr}, Y_{tr}), \widetilde{N}$)
2: Initialize $\beta = \phi$, and $H = \phi$
3: for j = 1 to $N$ do
4:   for i = 1 to $\widetilde{N}$ do
5:     Initialize $\mathbf{w}_i$ and $b_i$
6:     $h_i(x) \leftarrow$ calculate $h_i(x)$ by evaluating each g(x)
7:     $\beta_i \leftarrow$ calculate $\beta_i$
8:   end for
9:   return $h(x_j), \beta$
10:   $H \leftarrow H \cup h(x_j)$
11: end for
12: Update $\beta \leftarrow \beta$ by Eq. (9)
13: return $H, \beta$
14: Predict on $(X_{test})$ to get $v$ and $\theta$
15: Determine other OPF variables by using Eqs. ()()()(4)–(6)
```

3. Case study

To evaluate the efficiency of the proposed learning-augmented approach for OPF, two transmission networks at different scales (i.e., 500-bus and 4918-bus as summarized in Table 1) are selected. The

Table 1
500-bus and 4918-bus test networks selected from the GO competition.

Network	No. of generators	No. of switched shunt	No. of load bus	No. of branch
500-bus	90	17	200	599
4918-bus	1,340	486	2,558	6,727

network parameters are collected from the 'GO' competition repository [38].

3.1. Dataset generation

3.1.1. Input load dataset generation

The machine learning model training requires a significant amount of data. Generating training data from conventional optimization-based method has been a common practice in the literature [19,22,23,39–41]. In [22,23], IEEE-14 bus, 57-bus, 118-bus, and 300-bus systems were solved by MATPOWER for uniformly distributed random perturbations of the load at each bus. In addition, [19] also solved EPRI 39-bus, IEEE RTS 73-bus, RTE pegase 89-bus, IEEE dtc 162-bus, and edin 189-bus systems from the NESTA library [42] by introducing a similar load variation technique. In this work, to generate a sufficient amount of training data, the given base case of load distribution in the 'GO' competition is used as a benchmark. Sample load profiles are generated based on the hourly load profiles of the Electric Reliability Council of Texas (ERCOT) system from January 01, 2004 to July 15, 2009 [43]. The generated load profiles can represent the load variation characteristics (e.g., daily, weekly, monthly, and seasonal) in real power systems.

Since the test case load and ERCOT's load are at different magnitude scales, the ERCOT hourly load profile is scaled down for the 500-bus test network and scaled up for the 4918-bus test network. Regarding the load distribution, two scenarios are considered in this paper:

- Scenario 1: The scaled bus load profiles are used to test the learning-augmented OPF method.
- Scenario 2: The scaled bus load is added by Gaussian noises with a zero mean and 5% standard deviation.

The load profile generation method ensures the load variability in real networks. Due to the similarity of load profiles in consecutive years, there are a large number of repeating load values in the training dataset that may cause overfitting issues. Adding Gaussian noises in Scenario 2 seeks to solve the overfitting challenge.

3.1.2. OPF solution dataset generation

Based on the aforementioned load data generation scenarios, 10,000 and 40,000 load samples are produced for the 500-bus and the 4918-bus test networks, respectively. The generated load profiles are then fed into MATPOWER [44] to get the AC OPF solution. Since controllable shunts cannot be modeled directly in MATPOWER, they are modeled as controllable synchronous condensers. The MATPOWER 'MIPS' is adopted to solve the AC OPF due to its superior speed and accuracy. Real and reactive power losses for each scenario are also recorded for further checking of the nodal power mismatch. To reduce the burden on input–output mapping, voltage angles are converted to radians, which helps to improve the prediction accuracy.

3.2. Data cleaning

The developed learning-augmented OPF algorithm does not have the capability to discriminate between feasible and infeasible cases. Among all the load inputs, not all of them could provide feasible solution and only the solutions of converged cases are used for ML model training and testing. Cases where the MIPS solver doesn't produce a feasible solution are also excluded. To further reduce the mapping task, the angle of slack bus is excluded as it is taken as a reference when solving OPF. For some instances, the demand of some load buses remains constant, which does not affect the ML performance and are thus excluded from the input variable set.

3.3. Hyperparameter tuning

Though each ML model provides a wide range of hyperparameters, not all of them affect the model performance equally under certain experimental setups. Thus in the case study, the selection ranges for hyperparameters are narrowed down by considering practical constraints. A representative part of the whole dataset is selected for optimizing hyperparameter settings.

Fig. 2 shows the hourly load pattern of ERCOT and a yearly pattern is observed. Thus, the first 8,760 data points are selected for hyperparameter tuning, which are split in a 80%-20% ratio for training and validation. The parameter settings that produce the most consistency between training and validation are selected.

3.3.1. RF model

For the RF model, the following parameter settings are explored and compared, including the number of features to consider (`max_features=[0.3,0.4,0.5]`), the maximum tree depth (`max_depth=[10,20,30]`), the minimum number of samples required at each leaf node (`min_samples_leaf=[2,3,4]`), the number of trees (`n_estimators=[20,40,60,80,100]`), and the minimum number of samples required to split an internal node (`min_samples_split=[2,5,10]`). It is found that there exists an asymptotic relationship between the number of trees and prediction accuracy. The effects of other hyperparameters on the model performance are not readily conceivable and thus require empirical evaluation. Table 2 summarizes the final hyperparameter settings for the RF model.

3.3.2. DT model

The hyperparameters for the multi-target DT model are similar to those of the RF model, except that the DT model doesn't have the number of trees since it is a single tree multi-target regression model. Table 2 summarizes the final hyperparameter settings for the DT model.

3.3.3. ELM model

Compared to RF and DT models, ELM has fewer parameters to tune. Among them, only three parameters are observed to affect the model performance significantly. They are the number of units or neurons to generate the mapping in the hidden layer of 'SLFN' (`n_hidden= [256, 400, 512, 1024, 2048]` for the 500-bus network and `[4096, 5116, 6144,7168]` for the 4918-bus network), the mixing coefficient for distance and dot products of input activation functions (α = `[0.1,0.2,0.5]`), and the activation function. Since the reactive load q_d in the input dataset often takes a negative value, a `'sigmoid'` activation function is selected. Table 2 summarizes the final hyperparameter settings for the ELM model.

Fig. 3 shows the training and validation mean absolute errors (MAE) of the three ML models, changing with the single most influential hyperparameter. The values at which both the training and validation

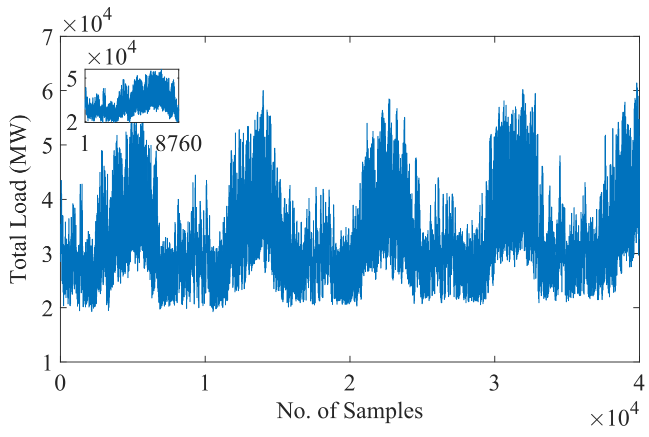

Fig. 2. ERCOT hourly total load profile (from January 01, 2004 to July 15, 2009); the inset shows the first representative 8,760 load samples, used for hyperparameter tuning.

Table 2
Optimal model hyperparameters.

Model	Network	Hyperparameter
RF model	500-bus	family n_estimators = 60
		family min_samples_leaf = 4
		family min_samples_split = 10
		family max_features = 0.4
		family max_depth = 10
	4918-bus	family n_estimators = 80
		family min_samples_leaf = 4
		family min_samples_split = 5
		family max_features = 0.3
		family max_depth = 30
DT model	500-bus	family max_depth = 30
		family min_samples_leaf = 2
		family min_samples_split = 2
		family max_features = 0.5
	4918-bus	family max_depth = 50
		family min_samples_leaf = 10
		family min_samples_split = 5
		family max_features = 0.3
ELM model	500-bus	family n_hidden = 2048
		family rbf_width = 0.5
		family $\alpha = 0.2$
	4918-bus	family n_hidden = 6096
		family rbf_width = 0.4
		family $\alpha = 0.1$

errors are consistent, are selected for final model training. The hyperparameter tuning of RF and DT models was performed by using Scikit-learn[1], and for ELM model tuning, D. Lambert's 'Python-ELM v0.3'[2] was used. Once the hyperparameter settings are finalized, the whole dataset is split into an 80%-20% ratio for training and testing.

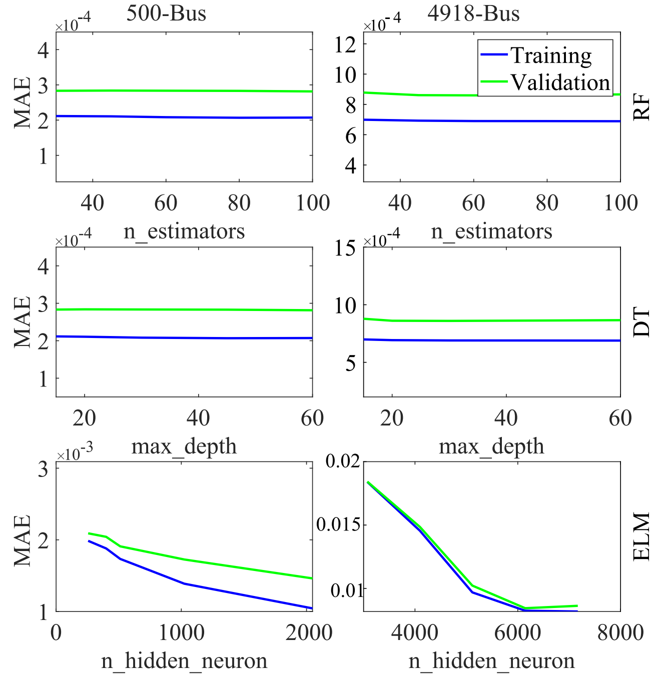

Fig. 3. Performance of ML models with the most significant hyperparameters in terms of MAE of the aggregated voltage magnitude and angle prediction (dimensionless).

[1] https://scikit-learn.org/stable/
[2] https://github.com/dclambert/Python-ELM

4. Results & discussion

4.1. Performance of ML models

The efficiency of the proposed learning-augmented approach heavily relies on the end-to-end prediction of bus voltage magnitude and angle. Predictions are performed on 1,240 converged test samples for the 500-bus network and 6,041 test samples for the 4918-bus network. Figs. 4 and 5 show the quality of the voltage magnitude prediction results compared with the actual magnitude obtained from MATPOWER. For the 500-bus test case, the lower and upper limits of the voltage magnitude are 0.9 and 1.1 per unit (p.u.), respectively; for the 4918-bus system, the bound is narrower ranging between 0.95 to 1.05 p.u.

It is observed that the RF and DT models perform better than the ELM model for the 500-bus test system. While most of the predicted voltage magnitudes of ELM are well within the technical bounds in Scenario 1, some violations are observed in Scenario 2. For the 4918-bus test system, all three models' results scatter around the diagonal line, and most of the predictions are well within their ranges even though the bound is more restricting. The deviation from the diagonal line can be attributed to the increase of mapping dimension. Figs. 6 and 7 show the voltage angle prediction results. Though the prediction is performed in radians for better prediction accuracy, it is plotted in degree for better observability. For both test networks and both scenarios, the lower and upper limits of angles are -180° and +180° ($-\pi$ and $+\pi$ in radian), respectively. It is observed that the predicted voltage angles from all three ML models are well within their limits for both test networks. The results of the 4918-bus network are slightly worse than those of the 500-bus network, due to its high dimensionality. Since both voltage magnitude and angle prediction results are within limits in most instances, severe constraints violation could be prevented.

While all the three ML models perform well, the two tree-based models (i.e., RF and DT) perform better than ELM. This can be attributed to the superior multi-target handling capacity of tree-based models, whereas the automatic feature selection based neural network approach does not present a clear advantage in case studies. A detailed description on the statistical performance of ML models is summarized in Tables 3 and 4 based on three metrics, including the mean absolute error (MAE), root mean square error (RMSE), and R-squared (R^2). The results in Scenario 2 is inferior to those in Scenario 1, which is expected due to the additional 5% variation in the dataset. But overall, the performance of the learning-augmented method is still reasonable in Scenario 2, especially with the RF model.

4.2. Post-processing from physics-based network equations

After solving the voltage magnitude and angle with ML models,

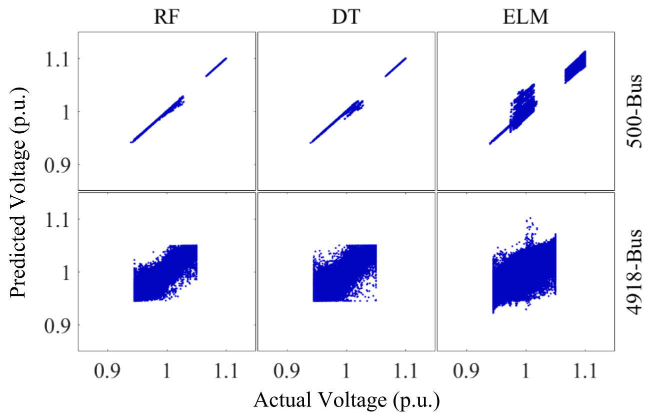

Fig. 4. Voltage magnitude comparison in Scenario 1: predicted *vs.* actual value from MIPS.

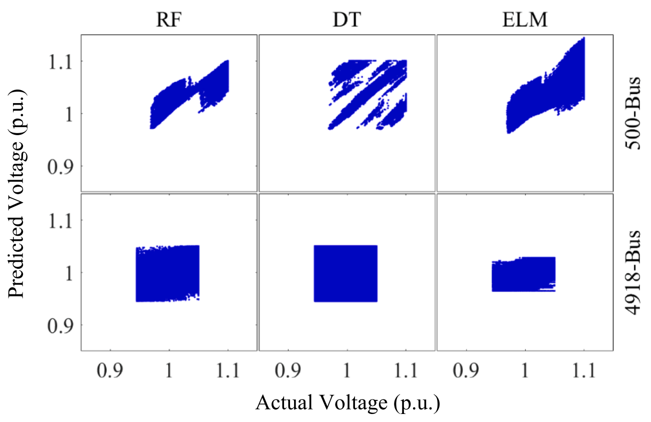

Fig. 5. Voltage magnitude comparison in Scenario 2: predicted *vs.* actual value from MIPS.

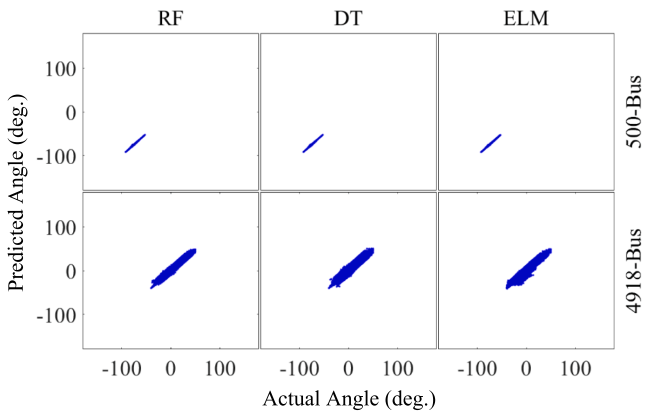

Fig. 6. Voltage angle comparison in Scenario 1: predicted *vs.* actual value from MIPS.

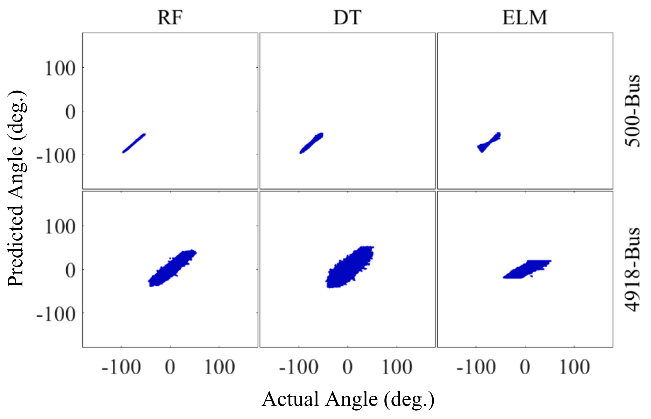

Fig. 7. Voltage angle comparison in Scenario 2: predicted *vs.* actual value from MIPS.

branch power flows are calculated and shown in Figs. 8 and 9. It is observed that predicted branch power flows closely follow actual branch flows in Scenario 1 but slightly deviate in some models of Scenario 2. For the 500-bus test network, the hard-limit of the branch power flow is relaxed for some instances by MIPS, which is reflected in the results. For the 4918-bus test network, the relaxation is only required in few cases and the actual and predicted branch flows are within their usual operating limits in most of the cases. This branch flow constraint relaxation allows the MIPS solver to have higher voltage angle variation, which is

Table 3
Statistical performance comparison for the voltage magnitude prediction.

Model	Network	Scenario	MAE (p.u.)	RMSE (p.u.)	R^2
		1	7.73×10^{-5}	0.0005	0.9998
RF	500-bus	2	0.0024	0.0076	0.9576
		1	0.0001	0.0070	0.9999
	4918-bus	2	0.0017	0.0050	0.9369
		1	3.92×10^{-5}	0.0003	0.9999
DT	500-bus	2	0.0039	0.0140	0.8575
		1	5.28×10^{-5}	0.0006	0.9952
	4918-bus	2	0.0021	0.0069	0.8216
		1	0.0023	0.0050	0.9811
ELM	500-bus	2	0.0082	0.0127	0.8844
		1	0.0008	0.0022	0.9521
	4918-bus	2	0.0068	0.0112	0.7124

Table 4
Statistical performance comparison for the voltage angle prediction.

Model	Network	Scenario	MAE (deg.)	RMSE (deg.)	R^2
		1	0.0005	0.0015	0.9945
RF	500-bus	2	0.0021	0.0036	0.9872
		1	0.0013	0.0080	0.9906
	4918-bus	2	0.0160	0.0380	0.8773
		1	0.0003	0.0013	0.9969
DT	500-bus	2	0.0043	0.0085	0.9396
		1	0.0013	0.0102	0.9866
	4918-bus	2	0.0227	0.0586	0.8252
		1	0.0017	0.0030	0.9834
ELM	500-bus	2	0.0093	0.0196	0.8115
		1	0.0072	0.0160	0.9656
	4918-bus	2	0.0580	0.0891	0.7521

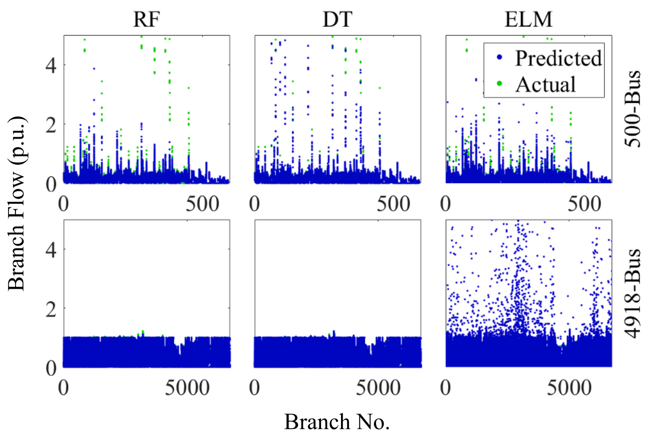

Fig. 8. Branch power flow profiles in Scenario 1 (the power flow of each branch is scaled to its p.u. capacity).

illustrated in Fig. 10.

The violations of voltage parameters and branch flows are summarized in Table 5. It is observed that the RF-based model presents fewer branch flow violations than the benchmark MIPS, for both test networks and both scenarios.

In the context of real power system operation, it is not required to execute a perfect balance between the generation and consumption due to the inherent uncertainty in their nature. Since the real power system operation relies on an automatic generation control (AGC) mechanism responding to area control error (ACE) to bridge this gap [45], dynamic simulations are required, which is beyond the scope of this work. Considering the high accuracy of the prediction compared with the benchmark solver, the learning-augmented OPF approach would not affect the reliability of the power system.

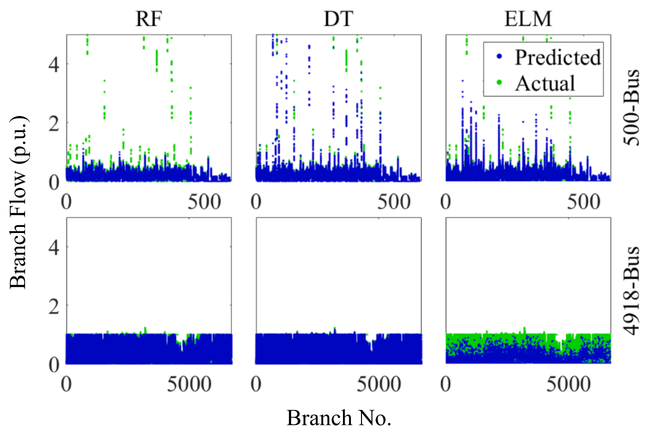

Fig. 9. Branch power flow profiles in Scenario 2 (the power flow of each branch is scaled to its p.u. capacity).

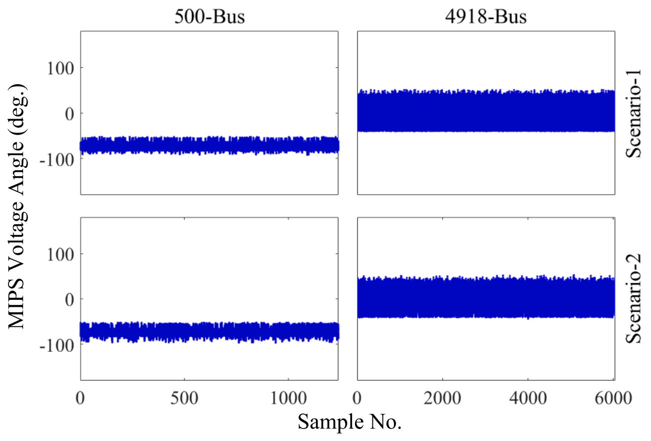

Fig. 10. Voltage angle variation in the MIPS result (actual angle)

Table 5
Solution quality comparison in terms of feasibility.

Solver	Network	Scenario	Voltage violation (%)	Angle violation (%)	Branch flow violation (%)
MIPS	500-bus	1	0	0	0.15
		2	0	0	0.20
	4918-bus	1	0	0	0.14
		2	0	0	0.11
RF	500-bus	1	0	0	0.02
		2	0	0	0
	4918-bus	1	0	0	0.10
		2	0	0	0.01
DT	500-bus	1	0	0	0.16
		2	0	0	0.21
	4918-bus	1	0	0	0.13
		2	0	0	0.06
ELM	500-bus	1	0.002	0	0.0009
		2	1.112	0	0.05
	4918-bus	1	0.060	0	0.19
		2	0	0	0

Table 6
Computational time comparison between the learning-augmented OPF solver and MIPS.

Solver	Network	Scenario	Average solution time (s)	Speedup factor	Training time (s)
MIPS	500-bus	1	0.55	–	–
		2	0.61	–	–
	4918-bus	1	10.53	–	–
		2	14.50	–	–
RF	500-bus	1	0.038	14.47	1,500
		2	0.040	15.25	1,800
	4918-bus	1	0.113	92.77	35,000
		2	0.160	90.62	55,000
DT	500-bus	1	0.032	17.18	700
		2	0.043	14.12	890
	4918-bus	1	0.110	95.72	19,000
		2	0.176	82.38	28,000
ELM	500-bus	1	0.035	15.71	4.48
		2	0.044	13.86	6.82
	4918-bus	1	0.150	70.20	725.28
		2	0.213	68.07	920

Table 7
Optimality results of learning-augmented OPF approaches.

Network	Scenario	RF	DT	ELM
500-bus	1	2.196×10^{-4}	3.516×10^{-4}	5.028×10^{-4}
(P = 1240)	2	8.153×10^{-4}	0.0338	0.0112
4918-bus	1	3.000×10^{-3}	3.126×10^{-3}	3.729×10^{-3}
(P = 6041)	2	6.526×10^{-4}	1.329×10^{-4}	5.490×10^{-4}

4.3. Computational time and solution quality comparison

Computational efficiency is the main driving factor to employ ML to solve AC OPF. To evaluate the computational advantage of the proposed learning-augmented method, the solving time between the proposed method and MIPS is compared in Table 6. Both scenarios were simulated on a high-performance computing facility using 40 processors and 64-GB memory.

Since the numbers of samples considered in both test networks are different, the average solution time per load instance is considered instead of the total solution time. The average solution time also represents the computational efficiency of solving one AC OPF instance. For learning-augmented approaches, the solution time consists of both ML prediction and network equation post-processing time. The post-processing computational time is similar between Scenarios 1 and 2. The speedup factor of the ML model is calculated based on the benchmark computational time of MIPS. It is observed that learning-augmented OPF models are approximately 15–20 times and 70–100 times faster than MIPS for the 500-bus and 4918-bus networks, respectively. Regarding the training time of ML models, the ELM model is significantly faster than RF and DT models.

To measure the solution quality of the proposed approach, the optimality checking metric O [23] is adopted.

$$O = \frac{1}{P}\sum_{n=1}^{P}\frac{c_g(Proposed) - c_g(Actual)}{c_g(Actual)} \tag{10}$$

where P is the total number of considered cases, which is equal to the number of originally converged samples in the test dataset. The optimality of the proposed approach on both networks, as summarized in

Table 7, is regarded in an acceptable range compared to other results reported in the literature [23].

5. Conclusion

This paper developed a learning-augmented method for solving AC optimal power flow, which integrated both power network equations and an MIMO ML model to yield a near-optimal AC OPF result. Three ML methods were employed and compared, including the random forest, multi-target decision tree, and extreme learning machine. The integration of network equations could ensure the satisfaction of inherent physical laws in the OPF problem. Results on both 500-bus and 4918-bus test networks have shown minimal constraints violation and loss of optimality. It was also found that the learning-augmented method could improve the OPF computational efficiency by approximately 15–100 times depending on the network size. Potential future work will explore the concept of 'learning to optimize' [46] to further improve AC OPF solution feasibility by learning the evolutionary pattern in optimization.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work was supported by the U.S. Department of Energy under Contract No. DE-AR0001075.

References

[1] Cain MB, O'neill RP, Castillo A. History of optimal power flow and formulations optimal power flow paper 1. In: Optimal power flow and formulation papers, Federal Energy Regulatory Commission (FERC); 2012.
[2] CAISO, Fifth Replacement Electronic Tariff; 2018.
[3] E.I.A. (US), Annual Energy Outlook 2012: With Projections to 2035, Government Printing Office; 2012.
[4] Coffrin C, Van Hentenryck P. A linear-programming approximation of ac power flows. INFORMS J Comput 2014;26(4):718–34.
[5] Calafiore GC, El Ghaoui L. Optimization models. Cambridge University Press; 2014.
[6] Low SH. Convex relaxation of optimal power flow–part i: Formulations and equivalence. IEEE Trans Control Network Syst 2014;1(1):15–27.
[7] Abido M. Optimal power flow using particle swarm optimization. Int J Electr Power Energy Syst 2002;24(7):563–71.
[8] Bakirtzis AG, Biskas PN, Zoumas CE, Petridis V. Optimal power flow by enhanced genetic algorithm. IEEE Trans Power Syst 2002;17(2):229–36.
[9] Wang Y, Wang S, Wu L. Distributed optimization approaches for emerging power systems operation: A review. Electric Power Syst Res 2017;144:127–35.
[10] Deka D, Misra S. Learning for dc-opf: Classifying active sets using neural nets. 2019 IEEE Milan PowerTech, IEEE 2019:1–6.
[11] Ng Y, Misra S, Roald LA, Backhaus S. Statistical learning for dc optimal power flow. 2018 Power Systems Computation Conference (PSCC), IEEE 2018:1–7.
[12] Guha N, Wang Z, Wytock M, Majumdar A. Machine learning for ac optimal power flow, arXiv preprint arXiv:1910.08842.
[13] Jamei M, Mones L, Robson A, White L, Requeima J, Ududec C. Meta-optimization of optimal power flow, ICML, Climate Change: How Can AI Help.
[14] Canyasse R, Dalal G, Mannor S. Supervised learning for optimal power flow as a real-time proxy. 2017 IEEE power & energy society innovative smart grid technologies conference (ISGT), IEEE 2017:1–5.
[15] Navidi T, Bhooshan S, Garg A. Predicting solutions to the optimal power flow problem. In: Project Report. Stanford University; 2016.
[16] Hasan F, Kargarian A, Mohammadi J. Hybrid learning aided inactive constraints filtering algorithm to enhance ac opf solution time, arXiv preprint arXiv: 2006.01336.
[17] Robson A, Jamei M, Ududec C, Mones L. Learning an optimally reduced formulation of opf through meta-optimization, arXiv preprint arXiv:1911.06784.
[18] Pan X, Zhao T, Chen M. Deepopf: Deep neural network for dc optimal power flow. In: 2019 IEEE International conference on communications, control, and computing technologies for smart grids (SmartGridComm), IEEE; 2019. p. 1–6.
[19] Fioretto F, Mak TW, Van Hentenryck P. Predicting ac optimal power flows: Combining deep learning and lagrangian dual methods. In: Proceedings of the AAAI conference on artificial intelligence 2020;34: p. 630–7.
[20] Sanseverino ER, Di Silvestre M, Mineo L, Favuzza S, Nguyen N, Tran Q. A multi-agent system reinforcement learning based optimal power flow for islanded microgrids. In: 2016 IEEE 16th international conference on environment and electrical engineering (EEEIC), IEEE; 2016. p. 1–6.
[21] Gutierrez-Martinez VJ, Cañizares CA, Fuerte-Esquivel CR, Pizano-Martinez A, Gu X. Neural-network security-boundary constrained optimal power flow. IEEE Trans Power Syst 2010;26(1):63–72.
[22] Baker K. Learning warm-start points for ac optimal power flow. 2019 IEEE 29th international workshop on machine learning for signal processing (MLSP), IEEE 2019:1–6.
[23] Zamzam A, Baker K. Learning optimal solutions for extremely fast ac optimal power flow, arXiv preprint arXiv:1910.01213.
[24] Shi D, Tylavsky DJ. A novel bus-aggregation-based structure-preserving power system equivalent. IEEE Trans Power Syst 2014;30(4):1977–86.
[25] Yang Z, Zhong H, Xia Q, Kang C. Fundamental review of the opf problem: Challenges, solutions, and state-of-the-art algorithms. J Energy Eng 2018;144(1): 04017075.
[26] Lavaei J, Low SH. Zero duality gap in optimal power flow problem. IEEE Trans Power Syst 2011;27(1):92–107.
[27] Wang H, Thomas RJ. Towards reliable computation of large-scale market-based optimal power flow. In: 2007 40th Annual Hawaii international conference on system sciences (HICSS'07), IEEE; 2007. p. 117–117.
[28] Coffrin CJ. Arpa-e grid optimization competition, scopf overview, Tech. rep., Los Alamos National Lab.(LANL), Los Alamos, NM (United States); 2019.
[29] ARPA-E, Challenge 1 problem formulation, [Online] Available at: https://gocompetition.energy.gov/challenges/challenge-1/formulation, [Accessed: 30 Oct. 2019].
[30] Pedregosa F et al., sklearn.multioutput.multioutputregressor, [Online] Available at: https://scikit-learn.org/stable/modules/generated/sklearn.multioutput.MultiOutputRegressor.html, [Accessed: 20 Dec. 2020].
[31] Spyromitros-Xioufis E, Tsoumakas G, Groves W, Vlahavas I. Multi-target regression via input space expansion: treating targets as inputs. Mach Learn 2016;104(1): 55–98.
[32] Breiman L. Random forests, Vol. 45, Springer; 2001. p. 5–32.
[33] Ho TK. Nearest neighbors in random subspaces. In: Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR). Springer; 1998. p. 640–8.
[34] Linusson H. Multi-output random forests, in: Master's Thesis, University of Borås, School of Business and IT; 2013.
[35] Kocev D, Naumoski A, Mitreski K, Krstić S, Džeroski S. Learning habitat models for the diatom community in lake prespa. Ecol Model 2010;221(2):330–7.
[36] Blockeel H, De Raedt L, Ramon J. Top-down induction of clustering trees, arXiv preprint cs/0011032.
[37] Huang G-B, Zhu Q-Y, Siew C-K. Extreme learning machine: theory and applications. Neurocomputing 2006;70(1–3):489–501.
[38] ARPA-E, Datasets, [Online] Available at: https://gocompetition.energy.gov/challenges/22/datasets, [Accessed: 30 Oct. 2019].
[39] Lei X, Yang Z, Yu J, Zhao J, Gao Q, Yu H. Data-driven optimal power flow: A physics-informed machine learning approach, arXiv preprint arXiv:2006.00544.
[40] Chen Y, Zhang B. Learning to solve network flow problems via neural decoding, arXiv preprint arXiv:2002.04091.
[41] Rahman J, Feng C, Zhang J. Machine learning-aided security constrained optimal power flow. IEEE Power Energy Soc General Meet, IEEE 2020.
[42] Coffrin C, Gordon D, Scott P. Nesta, the nicta energy system test case archive, arXiv preprint arXiv:1411.0359.
[43] Anon, Hourly load data archives, [Online] Available at: http://www.ercot.com/gridinfo/load/load_hist/, [Accessed: 30 Oct. 2019].
[44] Zimmerman RD, Murillo-Sánchez CE, Thomas RJ. Matpower: Steady-state operations, planning, and analysis tools for power systems research and education. IEEE Trans Power Syst 2010;26(1):12–9.
[45] Wood AJ, Wollenberg BF, Sheblé GB. Power generation, operation, and control. John Wiley & Sons; 2013.
[46] Evans GW, McGough B. Learning to optimize, Tech. rep., Citeseer; 2009.