Efficient Optimal Power Flow Flexibility Assessment: A Machine Learning Approach

Wengeng Pan, Chaoyue Zhao, Member, IEEE, Lei Fan, Senior Member, IEEE, Shuai Huang, Member, IEEE

2023 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT) | 978-1-6654-5355-4/23/$31.00 ©2023 IEEE | DOI: 10.1109/ISGT51731.2023.10066430

Abstract—We propose a framework based on machine learning to assess the flexibility of power systems with optimal power flow (OPF) model. The definition of flexibility is a range within which all demands are feasible. Conventional methods to evaluate the flexibility by solving a robust optimization problem are time-consuming for large-scale power systems. Machine learning provides us the opportunity to accelerate the computing process. We formulate the problem as a nonlinear binary classification problem and use a support vector machine (SVM) classifier with a Gaussian RBF kernel. To compute the flexibility, we solve a simple nonlinear equation based on the trained classification boundary. Then, we employ active learning to enhance the SVM's precision and adaptability. The simulation results for the five IEEE test cases indicate that our framework can compute the flexibility with a low error rate and significantly less execution time than the benchmark method for large-scale power systems.

Index Terms—Flexibility Assessment, Gaussian RBF Kernel SVM, Optimal Power Flow, Machine Learning.

I. Introduction

As a vital component of power system operations and management, optimal power flow (OPF) seeks to identify the optimal generation schedule to meet demand loads while minimizing the total generation cost under specific system operational constraints. Recent years have seen a rapid increase in the penetration of renewable energy sources (RES), which have introduced significant uncertainty into power system operations and posed significant challenges for independent system operators (ISOs) in maintaining system reliability and security. In face of such a high level of uncertainty and variability, ISOs need new tools to help schedule the resources in the system.

To evaluate a power system's ability to deal with the variability and uncertainty of net loads at reasonable cost, the concept of flexibility has been proposed. Several definitions of flexibility have been proposed and studied in relation to specific aspects of power systems [1]. For example, flexibility has been studied from system transmission and design's perspective [2], long-term generation planning's perspective [3], short-term power system operational perspective [4] and robust management in terms of economic dispatch (ED) and OPF's perspective [5], [6], [7]. In order to assess and quantify the flexibility of power systems under various definitions, the flexibility assessment problems are also studied. [5] examined the flexibility of power systems by explicitly concentrating on ED and dynamic automatic generation control (AGC). They formulated a robust optimization problem to determine the greatest variation of system loads that can be accommodated. To manage system ramp capacity, [7] proposed a robust optimization-based ED model and a lack of ramp probability index as an operational metric to evaluate system adaptability.

Although the flexibility assessment problems based on OPF or ED models have been studied in recent years, the majority of models are converted into computationally intensive and even intractable robust optimization problems[5], [8], [9]. However, machine learning (ML) provides us the chance to address this issue. The application of ML techniques to the OPF scheduling problem in order to speed up the computing procedure has gained increasing interest recently. For instance, [10] utilized deep neural network (DNN) in their frameworks to address a variety of OPF problem variants (e.g., DC-OPF, SCDCOPF and AC-OPF). They trained a DNN to learn the mapping between the input load and system operating decisions such as dispatch and power flow. Then for an arbitrary load input, the corresponding operating decision can be the output with the learned mapping. They demonstrated a significant computation time reduction by applying DNN as compared to conventional approaches. SVM was utilized in [11] to solve transient stability-constrained OPF (TSC-OPF) problems. The authors used SVM to discover the transient stability boundary and incorporated it into TSC-OPF as a constraint. By incorporating SVM into their proposed method, the training period can be drastically reduced. While machine learning techniques have been used to solve OPF operation problems, to the best of our knowledge, no literature has investigated machine learning's ability to address the OPF flexibility assessment problem.

In this paper, we propose a framework based on machine learning for evaluating flexibility in the OPF problem with the goals of achieving high evaluation accuracy and computational efficiency. We formulate the OPF flexibility assessment problem as a binary classification problem and classify the feasible and infeasible loads using Gaussian RBF kernel SVM [12]. Then, we iteratively resample and retrain SVM using active learning to obtain more accurate classification results. We then solve a simple equation to determine the flexibility metric's value. The key contributions of our paper can be summarized as follows:

- We propose a machine learning based framework for power system flexibility in OPF problems that can significantly reduce the computational time while maintaining excellent evaluation quality.
- We propose an iterative training and resampling pro-

Wengeng Pan, Chaoyue Zhao, and Shuai Huang are with University of Washington, E-mail:pwg15@uw.edu; cyzhao@uw.edu; shuaih@uw.edu. Lei Fan is with University of Houston, E-mail:lfan8@central.uh.edu. This research is funded by NSF ECCS-2046243 and ECCS-2045978.

cedure based on active learning to further improve the accuracy of the flexibility assessment and increase sample efficiency.
- We conduct case studies on five different IEEE test cases ranging from small scales to large scales to demonstrate the effectiveness of our proposed framework, particularly with respect to large scale systems.

The rest of this paper is organized as follows. Section III introduces the formulation of power system flexibility based on DC-OPF model and our proposed machine learning framework. Section IV conducts case studies on five IEEE standard test systems. Section V draws conclusions for our paper.

II. Model and Algorithms

In this section, we develop a framework to evaluate the power system flexibility based on DC-OPF model.

A. Flexibility Assessment based on DC-OPF

First, we briefly introduce the formulation to evaluate the flexibility of DC-OPF problem. Motivated by [5], we define the flexibility range as $P = [\bar{p}_d - \Delta p_d^{dn}, \bar{p}_d + \Delta p_d^{up}]$ that all net loads within this range can be tolerated by the system. That is, there exists at least one operation profile that can balance the net loads. Here Δp_d^{up} and Δp_d^{dn} are vectors which refer to the upper and lower net load deviation from the nominal net load $\bar{p}_d$ for each bus. In contrast to [5], which measures the total allowable deviation of all the buses in a composite manner, but is incapable of capturing power generation flexibility on any given bus, we measure the maximum relative net load deviation that the system can accommodate for all buses. Thus, the range ensures that all buses have a minimum level of net load uncertainty tolerance. Note that a net load outside of this range may not necessarily result in system infeasible operation. To obtain the maximum deviations Δp_d^{up} and Δp_d^{dn} for all buses, one can solve the following optimization problem:

$$\max_{\theta, p_g, \Delta p_d^{up}, \Delta p_d^{dn}} \quad \min_{i \in \{1,\cdots,n_d\}} \quad \Delta p_{d,i}^{up} + \Delta p_{d,i}^{dn} \tag{1}$$

$$\text{(Flex)} \quad \text{s.t.} \quad M_g p_g - M_d p_d = B_{bus}\theta, \quad \forall p_d \in P \tag{2}$$

$$p_{line}^{min} \le B_{line}\theta \le p_{line}^{max} \tag{3}$$

$$p_g^{min} \le p_g \le p_g^{max} \tag{4}$$

$$\Delta p_d^{up} \ge 0, \Delta p_d^{dn} \ge 0 \tag{5}$$

The objective function is to maximize the smallest range of power system net load among all buses within which the system can operate safely, where n_d is the number of load buses or length of vector p_d. Constraints (2) are the nodal power balance constraints which ensure the balance of power flow at each bus. M_g and M_d are matrices that map the generator and loads to the buses, p_g is the vector of decision variable of power generation. This should hold for any p_d within the flexibility range that we hope to find. Constraints (3) represent the limits on active power flow on each line, θ is the vector of voltage phase angle. Constraints (4) enforce power generation bounds for each generation resource.

This is a semi-infinite program that can be solved using the technique proposed by [5]. However, as the scale of the problem increases (which is common in real-world power system), the computational difficulty of the method in [5] increases significantly. To circumvent it, we propose our machine learning based framework to solve the flexibility assessment problem which can accelerate the computing process.

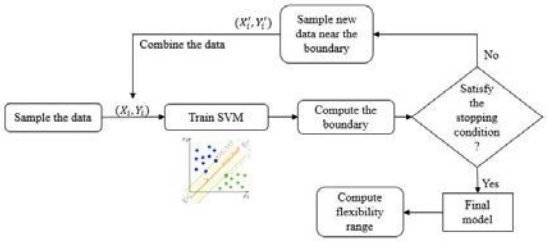

Fig. 1. Flowchart of flexibility assessment based on SVM

B. Framework Overview

Figure 1 provides a brief overview of the proposed framework. Solving the aforementioned flexibility assessment problem (1) is equivalent to searching for a hypercube of feasible power demand profiles (that we can find a generation schedule to fulfill the demand requirement). Instead of directly solving the optimization problem, we use SVM with Gaussian RBF kernel to find a approximated boundary between all the feasible and infeasible demands and then compute the desired set of feasible demand accordingly. Comparing with other machine learning models such as the DNN, the SVM is computationally easier to train and statistically more robust to initial values and other training parameters due to its convex formulation.

The first step in approximating the boundary is to collect training and testing demand data. We collect data samples using a uniform sampling method. Then, we assign a label to each sampled demand by solving a DC-OPF problem. The detailed procedure for sampling and labeling can be found in Section II-C.

Second, we normalize the sampled data and train a Gaussian RBF kernel SVM classifier. Typically, a power system has numerous buses; consequently, the dimension of the sampled demand is large. Due to the curse of dimensionality, even with a large sample size, the sampled data may still be sparse in high-dimensional space, which may result in an inaccurate classification model near the estimated boundary.

Active learning is a subset of machine learning in which new data points are queried and labeled interactively. The central concept of active learning is that a machine learning model can pick and choose from the training data to achieve greater accuracy [13]. The majority of active learning works can be categorized as uncertainty sampling [14], query-by-committee [15], [16], expected model change [17], estimated error reduction [18], etc.

Most these active learning algorithms assume that there is an existing unlabeled data set, and based on some criteria, the best one or group of new data is selected from the unlabeled

978-1-6654-5355-4/23/$31.00 ©2023 IEEE

data set and labeled for the subsequent training step. The selection process requires the training of multiple machine learning models or the resolution of computationally intensive optimization problems. In our case, we hope to obtain an accurate solution in a short amount of time, and we can query any point in a continuous space as opposed to being limited to a given unlabeled data set. Therefore, we propose an efficient and heuristic method for resampling data and querying their labels. In addition, we resample many new data points and query their labels simultaneously, as opposed to querying just one or a few new data points. After training the SVM with a Gaussian RBF kernel, we sample some new data near the estimated boundary using our resampling method, add them to the previously sampled data, and then train a new SVM. We repeat this procedure a predetermined number of times or until the estimated boundary no longer fluctuates significantly. Finally, we compute the variation range of the demand by solving a nonlinear equation based on the classification result. Details can be found in Section II-D.

C. Sampling and Labeling of Demand Data

We sample the total load data uniformly in a fixed range $[1_{n_g}^T p_g^{min}, 1_{n_g}^T p_g^{max}]$, where $1_{n_g}^T$ is a vector of length n_g (number of generators) with all elements being 1. When the sampled data has the total demand that is greater than the maximum generation capacity or less than the minimum generation capacity, it is infeasible naturally; therefore, we do not need to collect any data there. Then, we uniformly sample the demand on each bus with a sum equal to the total demand previously sampled. The sampled demand data is then fed into a DC-OPF solver to determine feasibility. If the demand is feasible, we label it as 1; otherwise, it is labeled as 0. For those data labeled as 1, constraints (2)-(5) are satisfied; for those labeled as 0, constraints (2)-(5) are not satisfied. We keep a total number of N_s of samples $X_1, \cdots, X_N$ with half of them feasible and half of them infeasible (note that X_i is an n_g-dimension vector). In addition, the sampled data can be represented as $(X_1, Y_1), \cdots, (X_{N_s}, Y_{N_s})$.

D. The Support Vector Machine

RBF kernel function is a widely used kernel function which can work with a general distribution of the data. It can help make proper separations when no prior knowledge of data is available [12]. Therefore for generality we use the SVM with Gaussian RBF kernel to build a binary classifier to classify the sampled data. Before passing the data into the classifier, we normalize the data into range [0, 1]. Then we train a Gaussian RBF kernel SVM as follows:

$$\min_{\lambda_1 \cdots, \lambda_{N_s}} \sum_{i=1}^{N_s} \lambda_i - \frac{1}{2} \sum_{i=1}^{N_s} \sum_{j=1}^{N_s} \lambda_i \lambda_j Y_i Y_j K(X_i, X_j) \quad (6)$$

$$\text{s.t.} \quad 0 \le \lambda_i \le C, \quad (7)$$

where $K(X_i, X_j) = \exp\left(-\gamma ||X_i - X_j||^2\right)$, λ_i are dual variables for SVM. The Gaussian RBF kernel performs well in classification problems that may have nonlinear patterns. When γ and C are chosen appropriately, the algorithm can efficiently learn the shape of the nonlinear boundary and achieve high classification precision. By solving the above SVM problem (6), we get $\lambda_1, \cdots, \lambda_{N_s}$. Most of them will be 0, and for i where $\lambda_i > 0$, we call X_i a support vector. The classification boundary of the trained SVM can be represented with $\lambda_1, \cdots, \lambda_{N_s}$ as:

$$\sum_{i:\lambda_i>0} \lambda_i Y_i K(x, X_i) + Y_m = \sum_{i:\lambda_i>0} \lambda_i Y_i K(X_m, X_i), \quad (8)$$

where m is an arbitrary support vector with $0 < \lambda_m < C$.

In the flexibility assessment problem (Flex), the uncertainty set is $P = [p_d - \Delta p_d^{dn}, p_d + \Delta p_d^{up}]$. It is easy to prove that the optimal solution to problem (Flex) should satisfy $\Delta p_{d,i}^{up} = \Delta p_d^{up}$ and $\Delta p_{d,i}^{dn} = \Delta p_d^{dn}$ for $i = 1 \cdots n_d$. Therefore it is equivalent to find a hypercube with equal length on each dimension where all demands in it are feasible. Then in the SVM problem it is equivalent to find $p_d + \epsilon_1 \times 1_{n_d}$ and $p_d - \epsilon_2 \times 1_{n_d}$ that is on the SVM classification boundary. We solve the equation (9) with scalar ϵ to achieve our goal:

$$\sum_{i:\lambda_i>0} \lambda_i Y_i K(\bar{p_d} + \epsilon 1, X_i) + Y_m = \sum_{i:\lambda_i>0} \lambda_i Y_i K(X_m, X_i). \quad (9)$$

In some cases, there will be more than two roots when solving this equation (9). We will only keep the positive root ϵ_+ and the negative root ϵ_- that are the closest to 0 based on the definition of P. Correspondingly, the flexibility will be $[p_d + \epsilon_- \times 1_{n_d}, p_d + \epsilon_+ \times 1_{n_d}]$.

The number of buses in a power system is typically very large, which can be more than 1,000. Therefore, the SVM classification is conducted in a space with a high number of dimensions. Even though we have a large sample size, such as 100,000 samples, they may be extremely sparse in the high-dimensional space, resulting in an inaccurate estimate of the boundary. In order to improve the precision of the classification boundary and the flexibility range, we sample more data around $\bar{p}_d + \epsilon_- \times 1_{n_d}$ and $\bar{p}_d + \epsilon_+ \times 1_{n_d}$.

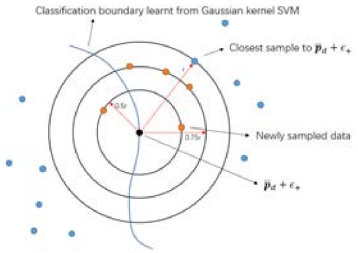

Fig. 2. Method to sample new data

Figure 2 depicts our sampling strategy in a two-dimensional space. The distance between the closest data in the old sample and $p_d + \epsilon_+ \times 1_{n_d}$ is denoted by r. Then, we know that there are no samples inside the ball with $p_d + \epsilon_+ \times 1_{n_d}$ as the center and r as the radius. Therefore, more samples should be collected from the ball. Due to the curse of dimensionality, uniformly sampling data inside the ball is a poor choice, as the sampled points are primarily located near the sphere of the ball. To obtain more new samples inside the ball, we uniformly sample new data on the sphere where $p_d + \epsilon_+ \times 1_{n_d}$ is the

978-1-6654-5355-4/23/$31.00 ©2023 IEEE

center and 0.75r is the radius. We also sample new data on the sphere with $\bar{p}_d + \epsilon_+ \times 1_{n_d}$ as center and 0.5r as radius. The newly sampled data will then be fed into a DC-OPF solver to obtain each data's label.

Then, we train an updated Gaussian RBF kernel SVM on the combined data set and compute an updated flexibility range. We repeat the procedure a predetermined number of times or until the change in flexibility range is negligible.

III. Case Study

In this section, we evaluate the performance of the our machine learning assisted flexibility assessment framework. The experiments are performed within Intel Core i7-8750H CPU and 16G memory.

TABLE I
Estimated flexibility values v.s. number of iterations and benchmark true flexibility values

num_iter	case 9	case 30	case 39	case 57	case 118
1	164.277	10.107	77.160	16.102	33.875
2	171.044	6.860	134.430	7.531	16.045
3	179.505	5.175	104.562	12.503	25.839
4	170.630	3.920	85.518	14.005	20.944
5	179.109	5.183	102.684	11.019	17.988
6	170.158	4.355	88.967	12.961	20.598
7	178.022	4.939	83.102	11.669	18.463
8	170.142	4.304	89.595	12.582	19.626
9	177.146	4.709	85.316	13.486	18.651
10	170.106	4.350	90.233	12.567	19.636
Benchmark	172.67	4.41	88.35	12.01	20.52

We sample 100,000 data as the original data, with 50,000 feasible and 50,000 infeasible demands. At each iteration, new points near the boundary are sampled according to the sampling method described in section II-D. We test our machine learning based framework on the IEEE 9-bus system, IEEE 30-bus system, IEEE 39-bus system, IEEE 57-bus system and IEEE 118-bus system. As a benchmark, we employ the traditional robust optimization method proposed by [5]. Our method is compared to the benchmark on two dimensions: computation time and assessment precision. In section III-A to III-C, we test the relationship of assessment precision with 3 factors: Number of initial sample size, number of iteration numbers and number of added sample size. We perform the experiments on 5 test cases. In section III-D we compare the relative error by using our sampling method and random sampling method. In section III-E we compare the computation time of our method and the benchmark method. Table I presents the true value of flexibility evaluated from the benchmark robust optimization method and estimated flexibility values for case 9, case 30, case 39, case 57 and case 118 with initial training sample size 1,000, and additional samples 10 in each iteration.

A. Impact of Initial Sample Size

In this subsection, we first investigate the relationship between number of initial sample size and the relative error. The relative error is defined as $\frac{|\Delta\tilde{p}_d^{up} - \Delta\hat{p}_d^{up}|}{\Delta\hat{p}_d^{up}}$, where $\Delta\tilde{p}_d^{up}$ is the upper flexibility computed by our proposed method, and $\Delta\hat{p}_d^{up}$ is the upper flexibility computed by the benchmark method. Here we assume that the benchmark method can find the ground truth flexibility of the power system. The number of initial sample size is set as 1,000, 5,000, 10,000, 15,000, 20,000, 25000 and 30,000 for five test cases respectively. The number of added samples is set as 10, and the number of iterations is 10. We run our experiment ten times for each number of sample size and calculate the mean and variance of the relative errors of the 10 experiments. We show the trend of the average relative error and the variance of the relative error in Figures 3. The X-axis represents the number of initial samples and the Y-axis represents the average relative error and variance of relative error. From Figures 3, we can see that the average relative error and the variance of the relative error does not change significantly as the number of initial sample changes.

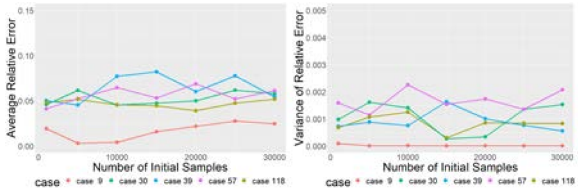

Fig. 3. Number of initial sample v.s. average relative error and variance of relative error

B. Impact of Iteration Numbers

In this subsection, we investigate the relation between iterations and error. The number of additional samples is set to 10, and the initial sample size is 1,000. Experiments are conducted for iterations 1 through 10. For each iteration, the experiment is repeated ten times. Figure 4 depicts the trend of the average relative error and the variance of the relative error. As the number of iterations increases, both the average error rate and the variance of the error rate exhibit a decreasing trend. After approximately ten iterations, the average and variance of error rate converge and remain relatively stable.

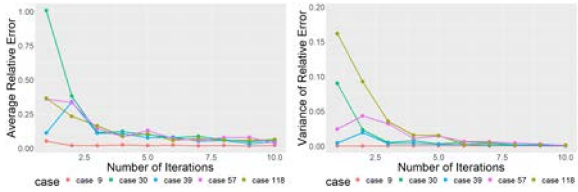

Fig. 4. Number of iterations v.s. average relative error and variance of relative error

C. Impact of Added Sample Size

In this subsection, we report the relationship between sample size addition and relative error. 1,000 is set as the initial sample size, and 10 is set as the number of iterations. The added sample size ranges from 10 to 100, with 10 serving as the interval. We repeat the experiments ten times for every additional sample size. Figures 5 depict the trend of the average relative error and the variance of the relative error for the 10 experiments. According to the data, the average relative error does not appear to have a strong correlation with the initial sample size. However, the variance decreases slightly as the number of samples increases.

978-1-6654-5355-4/23/$31.00 ©2023 IEEE

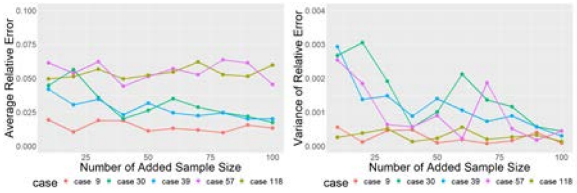

Fig. 5. Number of added sample size v.s. average relative error and variance of relative error

D. Impact of resampling method

In this subsection, we conduct experiments to demonstrate the effectiveness of our resampling procedure. We set the numbers of the initial sample as 1000, added sample per iteration as 10, and iterations as 10, respectively. We carry out the experiments by using our resampling method mentioned in section II-C in each iteration and by using uniform random sampling (which is the same as the initial dataset sampling) in each iteration. The experiments are run ten times for each sampling method.

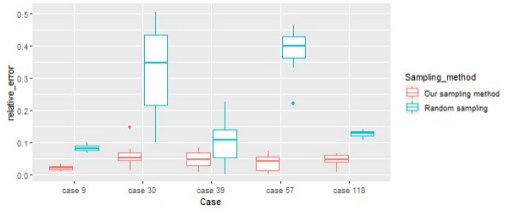

Fig. 6. Average relative error our resampling method and uniform random resampling method

Figure 6 shows the boxplots for the relative error of the five test cases under our resampling method and the random resampling approach. We can see that our method can outperform random sampling approach in terms of relative error.

E. Running time

The section III-A to III-C demonstrate that the number of iterations has a significant impact on the relative error. The number of additional samples has a minor effect on the relative error. The number of initial sample size has minimal effect on relative error. Therefore, we set the initial sample size to 1,000, the number of iterations to 10, and the added sample size to 50. We compare the running time of our proposed method with the benchmark robust optimization method [5] in Table II, where we report the total running time for the benchmark, and the total time to train the Gaussian RBF kernel SVM (6) and solve equation (9) for our method.

It can be observed that although our method does not outperform the benchmark robust optimization method for small-scale test cases, our method's execution time is significantly less than the benchmark for large cases.

TABLE II
RUNNING TIME (S)

	Proposed Framework	Benchmark
case 9	20.67	0.112
case 30	24.171	2.464
case 39	45.367	2933.9
case 57	32.515	1704.915
case 118	80.509	43512.184

IV. CONCLUSION

In this paper, we proposed a machine learning based framework to evaluate the power system flexibility with DC-OPF model. We classified feasible and infeasible demand loads using Gaussian-kernel SVM and then solved a nonlinear equation to obtain the uncertainty set and flexibility metric. Then, we proposed an efficient resampling method to sample new data and train a new Gaussian-kernel SVM in an iterative manner in order to achieve greater accuracy and reduce the number of samples used. Our case studies demonstrated that our method can achieve high assessment accuracy while reducing computation time significantly.

REFERENCES

[1] Jinye Zhao, Tongxin Zheng, and Eugene Litvinov. A unified framework for defining and measuring flexibility in power system. IEEE Transactions on Power Systems, 31(1):339–347, 2016.
[2] P Bresesti, A Capasso, MC Falvo, and S Lauria. Power system planning under uncertainty conditions. criteria for transmission network flexibility evaluation. In 2003 IEEE Bologna Power Tech Conference Proceedings, volume 2, pages 1–6. IEEE, 2003.
[3] Eamonn Lannoye, Damian Flynn, and Mark O'Malley. Evaluation of power system flexibility. IEEE Transactions on Power Systems, 27(2):922–931, 2012.
[4] Andreas Ulbig and Göran Andersson. On operational flexibility in power systems. In 2012 IEEE Power and Energy Society General Meeting, pages 1–8. IEEE, 2012.
[5] Lei Fan, Chaoyue Zhao, Guangyuan Zhang, and Qiuhua Huang. Flexibility management in economic dispatch with dynamic automatic generation control. IEEE Transactions on Power Systems, 37(2):876–886, 2022.
[6] Zhijun Qin, Yunhe Hou, Shunbo Lei, and Feng Liu. Quantification of intra-hour security-constrained flexibility region. IEEE Transactions on Sustainable Energy, 8(2):671–684, 2017.
[7] Anupam A Thatte, Xu Andy Sun, and Le Xie. Robust optimization based economic dispatch for managing system ramp requirement. In 2014 47th Hawaii International Conference on System Sciences, pages 2344–2352. IEEE, 2014.
[8] Dimitris Bertsimas, Eugene Litvinov, Xu Andy Sun, Jinye Zhao, and Tongxin Zheng. Adaptive robust optimization for the security constrained unit commitment problem. IEEE transactions on power systems, 28(1):52–63, 2012.
[9] Yu An and Bo Zeng. Exploring the modeling capacity of two-stage robust optimization: Variants of robust unit commitment model. IEEE transactions on Power Systems, 30(1):109–122, 2014.
[10] Xiang Pan, Tianyu Zhao, Minghua Chen, and Shengyu Zhang. Deepopf: A deep neural network approach for security-constrained DC optimal power flow. IEEE Transactions on Power Systems, 36(3):1725–1735, 2021.
[11] Huy Nguyen-Duc, Linh Tran-Hoai, and Huy Cao-Duc. An approach to solve transient stability-constrained optimal power flow problem using support vector machines. Electric Power Components and Systems, 45(6):624–632, 2017.
[12] Bernhard Schölkopf, Alexander J Smola, Francis Bach, et al. Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT press, 2002.
[13] Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.
[14] David D Lewis and William A Gale. A sequential algorithm for training text classifiers. In SIGIR'94, pages 3–12. Springer, 1994.
[15] H Sebastian Seung, Manfred Opper, and Haim Sompolinsky. Query by committee. In Proceedings of the 5th Annual Workshop on Computational Learning Theory, pages 287–294, 1992.
[16] Prem Melville and Raymond J Mooney. Diverse ensembles for active learning. In Proceedings of the 21st International Conference on Machine Learning, pages 584–591, 2004.
[17] Burr Settles, Mark Craven, and Soumya Ray. Multiple-instance active learning. In Advances in Neural Information Processing Systems, volume 20, pages 1289–1296, 2007.
[18] Nicholas Roy and Andrew McCallum. Toward optimal active learning through monte carlo estimation of error reduction. In Proceedings of the International Conference on Machine Learning (ICML), pages 441–448, 2001.

978-1-6654-5355-4/23/$31.00 ©2023 IEEE