

Thomas Navidi
Suvrat Bhooshan
Aditya Garg

Predicting Solutions to the Optimal Power Flow Problem

Abstract

This paper discusses an implementation of gradient boosting regression to predict the output of the optimal power flow problem for transmission networks to reduce the computation time. The optimal power flow problem is an important optimization to minimize the cost of operating a transmission network. The inputs are the load demanded at each node and the cost of generation. The outputs are the power and voltage of each generator in the network. We used the IEEE 30 bus network as our static network model, and load data from PJM. The ground truth solutions were solved using MATPOWER. Gradient boosting regression provided the highest accuracy based on the mean squared difference between each output to the true optimal solution. Over 90% of predictions were within 5% of the true solution. However, approximately 60% of predictions violated one of the network constraints. This is avoided by using the predicted solution as a starting point to the optimal power flow problem. A case study shows that with highly variable load due to renewable penetration, the computation time of running the optimal power flow this way is reduced by up to 30%.

Introduction

Our team has used machine learning techniques to reduce the computation time required to solve the optimal power flow problem for transmission networks. The optimal power flow problem is usually used to minimize the cost of operating a transmission network with X loads and Y generators. Many different people are interested in solving optimal power flow problems such as transmission system operators, industrial plant operators, and utility companies. Utility companies use the study to determine how much power each generator they operate should generate at some period of time, while system operators use the study to determine where in the network power needs to flow to provide electricity to the many loads without outages. Industrial plants often contain a large number of high power loads and generators which need to be coordinated using an optimal power flow problem. These problems often contain thousands of power lines and hundreds of controls. In addition, it is nonlinear and nonconvex. Solving this problem requires a large amount of computational power. The inputs to the problem is the cost of power generation and the power requested by each load. The output is the voltage and power generated by each generator.

A variety of techniques are currently implemented to solve these problems. The current state of the field is to utilize convex optimization techniques such as Semidefinite Programming (SDP) or Second Order Cone Programming (SOCP). These techniques can be seen in papers [1] and [2] respectively. Also, interior point methods can be used for smaller networks. Another common method is to approximate the system as a DC power flow. This creates a linear optimization problem that is much easier to solve. We propose using machine learning to estimate the solution of an optimal power flow problem using fewer computational resources than what is currently done. This is beneficial for both utility companies and researchers in the field. Very little work has been done on this before, as the recent increase in renewable penetration is just starting to make this strategy appealing.

There are a large number of available data sets to use. For example, many IEEE standard networks were designed for researchers to simulate optimal power flow problems. One such network is the IEEE 30 bus network, which can be found in [3]. This test network is an actual portion of the

American Electric Power System in the Midwest. It would be most beneficial for us to solve the optimal power flow problem under conventional methods and compare the computation time to our machine learning implementation. This will accomplish both providing a training data set for us as well as providing a benchmark to see the benefits of implementing a machine learning algorithm. The work will be supervised by Professor El Gamal.

Data Collection

Initially, various machine learning algorithms were tested on the small IEEE 30 bus test network in order to determine which algorithms perform best. The data obtained for this initial test comes from PJM Interconnection LLC, which is a regional transmission organization in the Midwest and Mid-Atlantic region of the United States. It is published for public use at the reference, [4]. The input data is the power demanded for each region in PJM's control every hour every day for the years 2014 and 2015. This makes 17,520 different data points. Each region is considered a bus in the IEEE 30 bus test network. The values were then normalized to match the output capacity of the generators on the IEEE 30 bus test network. The reactive power generated was estimated based on the load data provided by PJM, and randomized to produce an interesting set.

The next set of data is the cost function for each generator. This is a quadratic function with the form:

$$c = \lambda_1 P^2 + \lambda_2 P$$

Where c is the cost in dollars of generating P power from the generator. The total cost of operating the network is the sum of each generator's cost. The optimal power flow problem determines the power generated at each generator to minimize the cost, and the goal of our machine learning algorithm is to predict the power generated at each generator. The values of interest are the λ values for each generator. In the IEEE 30 bus test case there are 6 generators for a lambda vector of size 12. These values were determined by sampling a truncated Gaussian random variable from -3 to 3 with zero mean and variance 1. This random variable was scaled by 3 and divided by 10 to provide a percent deviation from the initial generator cost function values. The initial values come from the IEEE 30 bus test case.

The output we are trying to predict is the amount of power generated by each generator. This is found by solving the optimal power flow problem in MATLAB. The solver is a program called MATPOWER cited in [5]. It is specifically designed for researchers to use to solve the optimal power flow problem for transmission networks and uses an interior point method.

Features

Of the 30 buses, only 21 of them actually consume power while the other buses are transformers or other devices which do not consume real power. These 21 buses also consume reactive power. These, combined with the 12 lambda values, makes a total feature vector of 54 for the machine learning algorithm. The output vector is then size 18, which is the power and reactive power generated at each generator as well as the operating voltage of the generator. These 54 features combined with the static network are all that is needed to solve the optimal power flow problem conventionally. Figure 1 below shows the static test network used and an example load profile as an input feature.

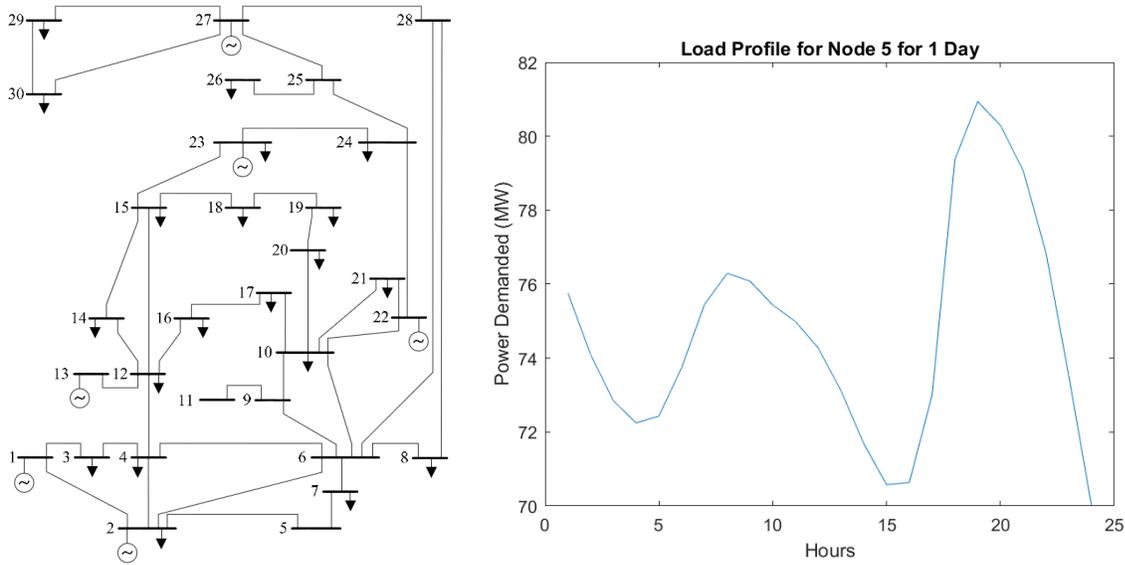


Figure 1: Test Network and Example Load Input Data Feature

Algorithms

We split the data samples randomly into 70% for training and 30% for testing. The first algorithm we implemented was stochastic gradient descent for linear regression in Python. This performed very poorly and was quickly abandoned. The next algorithm tested was support vector regression in Python. This algorithm was sub-optimal but still better than SGD. We then tried both Multi-layer Perceptron Regression with logistic loss and Gradient Boosting Regression (GBR) with Least Square Loss. Although both performed reasonably well, gradient boosting regression performed the best. The training and test errors are published in Table 1 below.

Table 1: Training and Test Errors for Various Implementations

Algorithm	Training Accuracy (Power)	Test Accuracy (Power)
Stochastic Gradient Descent with Least Square Loss	0%	0%
Support Vector Machine Regression with RBF Kernel	56.77%	54.87%
Multi-layer Perceptron Regression with Logistic Loss	96.51%	95.15%
Gradient Boosting Regression (GBR) with Least Square Loss	98.31%	98.8%

After picking Gradient Boosting Regression, we performed grid search to tune our hyperparameters. The optimal results were found at setting the number of estimators to 1000, learning rate to 0.1 and loss function to squared loss. We also tried building interaction terms from our existing feature set but did not see any significant improvement in performance.

Gradient Boosting is an additive model in a forward stage-wise fashion. It allows for the optimization of arbitrary differentiable loss functions. In each stage a regression tree is fit on the negative gradient of the given loss function. This algorithm is fairly robust to overfitting so a large number of estimators usually results in better performance. This works in our favour, as we are solving a well-

defined problem where neither the training nor test data has any outliers. Hence, the more we overfit the better our predictions.

Error Metric

For the error metric, we calculate the MSE, and divide it by the square of the optimum power generation to normalize it to get a value between 0 and 1. We calculate the average for each example, and report the average training and testing error. This is the form of the following equation:

$$Error = \frac{1}{m} \sum_{i=1}^m \frac{\|Y - P\|_2^2}{\|P\|_2^2}$$

Where P is the optimal power generation vector from the optimal power flow problem, Y is the predicted power generation from the machine learning implementation, and m is the number of training or test examples. This is done to get a separate error metric for real power, reactive power, and generator voltage. This is because the magnitude scales of the three variables are quite different, and it is good to compare the accuracy among the variables predicted.

Implementation and Results

After running hyperparameter tuning, we computed the maximum test accuracy using the error metric described above. This is shown in Table 1 below. Figure 2 shows the data error results and an example output superimposed on the ground truth. We were able to get such high accuracy due to the lack of outliers in the solutions as described above.

Table 1: Training and Test Errors for Gradient Boosting Regression

Prediction	Training Accuracy	Test Accuracy
Power	98.31%	98.8%
Reactive Power	98.11%	98.7%
Voltage	99.16%	99.43%

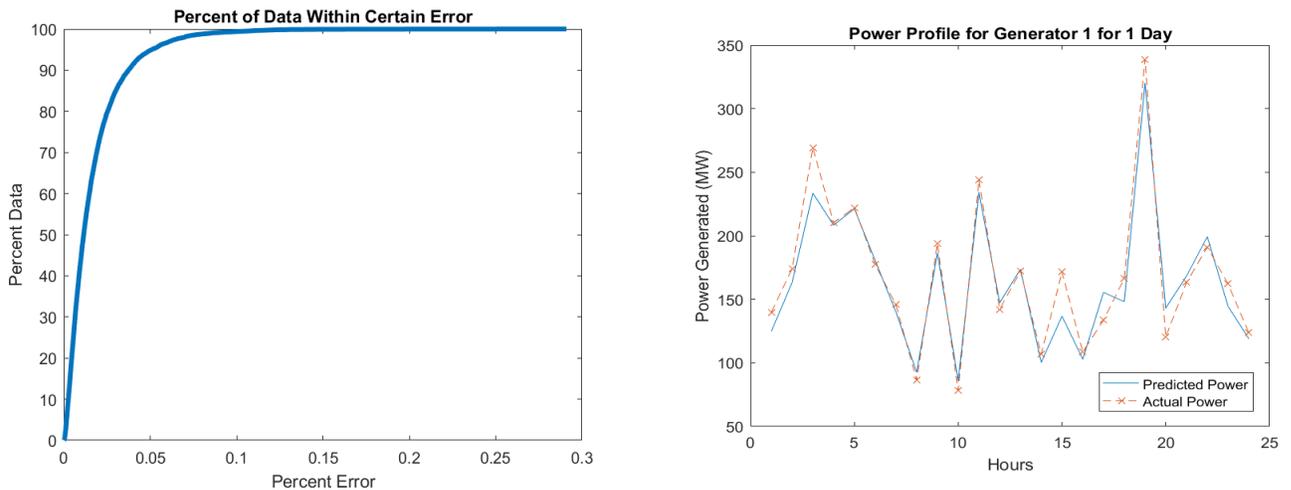


Figure 2: Percent of Data Within a Certain Error and Example Output Power Profile

Conclusions and Benefit

Although our algorithm was able to predict the power and voltage generated within high precision, some of the results violated the network constraints of the optimization problem. The constraints are the voltage must be within 6% of the nominal rating, and the power generated must equal the power demanded at all nodes. Approximately 60% of the predictions violated at least one constraint somewhere on the network. We tried forcing either constraint to be met in the output of our algorithm; however, since the power and voltage are not considered together, it was impossible to satisfy both simultaneously. For example, fixing the algorithm to only output valid voltages would satisfy the voltage constraints everywhere, but when power was calculated, it was found to violate several times. In the end, the benefit was negligible.

Instead of using the predicted values directly, they can be used as the starting point of the optimization problem. Having a very close starting point makes the algorithm converge in fewer steps improving computation time. Currently, the solution to the previous hour or 15 minutes is used as the starting point to the next. This is very accurate because the power demanded and generated does not change much within in time frame. However, with increased renewable penetration such as wind, power generated and demanded can become highly variable within this time frame. Table 2 below shows the computational improvement over both current load data and highly variable renewable energy data. As shown, a valuable improvement in computation time can be gained when energy sources are highly variable.

Table 2: Computation Time Improvement

Data	Previous	Prediction	Gain
Real 30 node data	699 sec	693 sec	0.85%
Highly Variable data	1153 sec	802 sec	30.44%

Future Work

The results we have are really good, and the models we have trained can be used to reduce computational times during simulations. However, our training data is only on 6 generators, and 21 loads, which is very small, and unrealistic. We would like to run them on much larger networks, and see how well they generalize. A larger scale implementation can be performed on a European 13,659 bus network. This network has 4,092 generators, which each need to have a predicted power generated. The computational time for optimizing a large system like this is much greater than the time for the IEEE 30 bus test case, so we will be able to compare computational time in this case. The IEEE 30 case converged to the optimal solution within milliseconds on a personal computer, so it does not provide a good benchmark for the computational time of typical optimal power flow problems.

We would also like to consider the dependency between power and voltage. Currently each output is independently predicted, but there is a quadratic relationship between voltage and power that should be explored. This could help reduce the number of constraint violations our predictions have. Finally, we would like to apply our results to a distribution network with distributed renewable generation and storage. This added complexity makes the traditional power flow prohibitively slow, but predicting a close solution would be immensely beneficial to storage control decisions.

References:

- [1] X. Bai, H. Wei, K. Fujisawa, and Y. Yang, "Semidefinite programming for optimal power flow problems," *International Journal of Electric Power & Energy Systems*, vol. 30, no. 6, pp. 383–392, 2008.
- [2] S. Huang; Q. Wu; J. Wang; H. Zhao, "A Sufficient Condition on Convex Relaxation of AC Optimal Power Flow in Distribution Networks," in *IEEE Transactions on Power Systems* , vol.PP, no.99, pp.1-1
- [3] Christie, Rich; Dabbaghi, Iraj. "30 Bus Power Flow Test Case." Power Systems Test Case Archive – University of Washington. Web. 20 Oct. 2016.
<https://www2.ee.washington.edu/research/pstca/pf30/pg_tca30bus.htm >.
- [4] PJM Interconnection LLC. "Metered Load Data." PJM Interconnection LLC. Web. 18 Nov. 2016.
<http://www.pjm.com/markets-and-operations/ops-analysis/historical-load-data.aspx>
- [5] R. D. Zimmerman, C. E. Murillo-Sánchez, and R. J. Thomas, "MATPOWER: Steady-State Operations, Planning and Analysis Tools for Power Systems Research and Education," *Power Systems, IEEE Transactions on*, vol. 26, no. 1, pp. 12-19, Feb. 2011.