Teaching Machines The Optimal Power Flow Problem

Thomas Navidi, Suvrat Bhooshan, Aditya Garg
Advised by: Dr. Abbas El Gamal

Introduction

**Goal:** Reduce computation time for solving the Optimal Power Flow (OPF) problem

OPF minimizes the cost of operating a transmission network which is a large multi-input multi-output nonlinear optimization problem. We tried to learn the optimal solution to reduce the computational time and resources needed to predict the optimal power generated for any given hour. Although, we achieved an average of almost 98% accuracy, the solution often violates voltage and power constraints. We used our predictions as a starting point for the OPF and reduced the computation time by almost 30%.

Data Model / Features

**Network:** IEEE 30 bus standard network from [1]

**Features:** Real and reactive power demanded and generation cost. Real power data is obtained from PJM [2] and processed to fit the network. Reactive power is a random variation of the real power data. Generation cost is a randomly generated vector based on actual costs from IEEE 30 bus case. Feature vector only 54 dimensional so used directly because they fully determine solution to the OPF with static network.

**Prediction:** Real and reactive power generated and generator voltage

**Ground Truth:** The solution to the OPF using MATPOWER interior point solver [3].

Algorithms

Algorithms we tried & the preliminary results:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Accuracy (Power)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stochastic Gradient Descent with Least Square Loss</td>
<td>0%</td>
</tr>
<tr>
<td>Support Vector Machine Regression with RBF Kernel</td>
<td>54.87%</td>
</tr>
<tr>
<td>Multi-layer Perceptron Regression with Logistic Loss</td>
<td>95.15%</td>
</tr>
<tr>
<td>Gradient Boosting Regression (GBR) with Least Square Loss</td>
<td>98.31%</td>
</tr>
</tbody>
</table>

- Hyperparameter tuning varying on GBR:
  - No. of Estimators (1000 estimators)
  - Learning Rate (alpha = 0.1)
  - Loss Function (Squared Loss)
- Tried polynomial feature interaction but no significant improvement in prediction accuracy.

Results

<table>
<thead>
<tr>
<th>Prediction</th>
<th>Training Accuracy</th>
<th>Test Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power</td>
<td>98.31%</td>
<td>98.8%</td>
</tr>
<tr>
<td>Reactive Power</td>
<td>98.11%</td>
<td>98.7%</td>
</tr>
<tr>
<td>Voltage</td>
<td>99.16%</td>
<td>99.43%</td>
</tr>
</tbody>
</table>

Table 1: Accuracy Results

<table>
<thead>
<tr>
<th>Data/OPF</th>
<th>Initialized to Zero</th>
<th>Initialized to Prediction</th>
<th>Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real 30 node data</td>
<td>699 sec</td>
<td>693 sec</td>
<td>0.85%</td>
</tr>
<tr>
<td>Highly Variable data</td>
<td>1153 sec</td>
<td>802 sec</td>
<td>30.44%</td>
</tr>
</tbody>
</table>

Table 2: Computation Time Improvement

Discussion

- ML predicts close to optimal solution
- Predictions almost instantaneous
- However, 60% of predictions violate constraints
  - Voltage varies more than 6%
  - Power generated doesn’t converge to optimal value. Producing more or less power is unacceptable.
  - Attempted to predict node voltages instead
    - Then compute power from voltage
    - Found that powers violate more in this case
    - Overall benefit is negligible
- Tried to incorporate constraints in loss function
  - Learning algorithm doesn’t know relationship between voltage and power, so cannot satisfy both simultaneously
- Running OPF initialized with predicted values converges to optimal solution faster. Predictions still provides benefit despite violations

Future Work

- Consider dependency between power and voltage
- Build a custom constraint forcing
  - Voltage < 6%
  - Power Produced ≥ Power Demanded
- While maintaining relationship between power and voltage
- Run a real size network (> 10,000 nodes)
- Apply to distribution networks with distributed renewable sources and battery storage

References

