EECS E6895 Advanced Big Data Analytics & Artificial Intelligence Final Report: Power Flow Optimization using Big Data Techniques

Rohan Raghuraman
Electrical Engineering
Columbia University
New York, NY
rr3417@columbia.edu

Abstract

The goal of this project is to optimize the power flow in a region under various conditions and to determine if the optimal power flow problem can be made faster using machine learning and AI. This is an important task because increased energy demand, addition of renewable energy and electric vehicles had led to increased challenges in terms of maintaining system stability and economic viability. Sub-optimal power flow conditions could exacerbate climate change. To tackle this task, firstly, load data of New England was scraped, followed by generating load data using different power flow simulations using Siemens PSS/E. Next the New England load data was fed into FBProphet to generate a prediction of future energy demand. A prediction of power flow was done by testing different supervised learning algorithms on the simulation data. Consequently, using domain expertise a determination was made that machine learning is a suitable alternative to perform power flow optimization, under certain circumstances. A dashboard was built to visualize the results. The end goal is that these techniques will contribute to lowering the time and computing energy needed to perform power flow optimization so that the development of the energy landscape will not be hindered.

1. Introduction

The objective of this paper is to optimize power flow in a given region under various conditions using machine learning and/or artificial intelligence. And if ML/AI can make the computing of power flow faster and less computationally-intensive. The formulation of power flow optimization is a challenging, computationally-intensive problem. Some objectives of power flow optimization include minimizing generator costs while adhering to physical constraints of power flow. Due to the constantly fluctuating nature of electricity demand, system operators must solve the AC optimal power flow problem as frequently as every five minutes over the entire grid [3]. Most traditional solutions for power flow optimization fail to converge within this time frame which is why the approximation, DC optimal power flow solution, is used. However, approximating solutions may exacerbate climate change by releasing unnecessary emissions, and could also cost billions of dollars in expenditure by having generators running when not needed [1]. Furthermore, the addition of renewables and electric vehicles into the grid will cause stability issues if power flow is not solved in real-time. Therefore, it is paramount that we find ways to optimize the AC power flow problem by other methods.

The following power flow equations have to be optimized:

$$P = \frac{|V_S| \quad |V_R|}{X_L} \quad \text{Sin}\, \delta$$

$$Q = \frac{|V_S| \quad |V_R|}{X_L} \left\{ \text{Cos}\, \delta - \frac{|V_S|}{|V_R|} \right\}$$

Figure 1. Power Flow Equations

The equations in Figure 1 show the active (P) and reactive (Q) power flow between a sending bus (S) and a receiving bus (R). The variables in the equation are the sending voltage (V_S) and receiving voltage (V_R). X_L is the reactance of the transmission line between the two buses, and δ is the power angle between the two buses. The variable of interest for us is the active power, P, because that is what is being used by the end customer [16]. The active power has to be kept at optimal value, i.e. at real-time energy demand. This project will aim to optimize the active power in a given region (New England) under various scenarios (varying demand and renewable energy penetration) using machine learning to determine if ML is a viable alternative to solve power flow to satisfy future energy demand. This is a cross-disciplinary project with knowledge and expertise required in the fields of computer science, optimization, and power systems.

2. Related Work

2.1. Machine Learning for OPF

With the amount of data that is being collected in power systems, machine learning is a powerful tool that can be leveraged to process this data to use in business decisions. Machine learning is already used in a few domains within the power systems field, specifically, to tackle the power flow optimization problem. Figure 2 shows the areas in which machine learning is used in OPF [4].

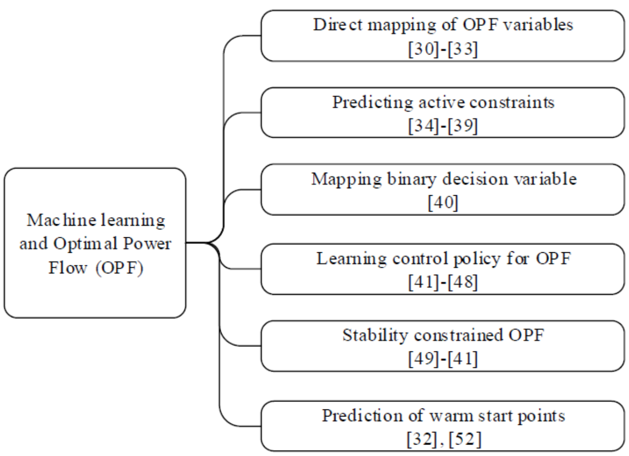

Figure 2. Machine Learning Applications for OPF

As noted in [4], machine learning is used in a variety of applications within the OPF field. For example, when trying to predict active constraints, Paper [8], presents an approach of using a set of active constraints at optimality to avoid having to directly map inputs to the optimal solution. This means that continuous mapping of inputs and outputs is no longer required, thereby, reducing computation time and cost. A streaming algorithm that ignores the problem structure, and learns the active sets from training samples of input parameters that correspond to the optimal solution is developed in this paper.

Another interesting application of ML to solve OPF is described in [4] as the direct mapping of OPF variables. Paper [14] details how a ”supervised learning-based security-constrained optimal power flow framework is developed which uses multi-target regression to map generation dispatching to local information”. A Pearson correlation plot is used to show the relationship of input features and targets which allows useless features to be ignored. A random forest model was used and found to be cheaper in terms of computational cost due to its ability to run in parallel. The prediction performance using local information was found to be acceptable by using local and global information.

2.2. Supervised Learning for OPF

Paper [2] studies the merits associated with approximation techniques for solving the power flow optimization problem. This paper only focuses on the cost function associated with OPF, which while being extremely relevant is something that this research project will not attempt to tackle. However, by using supervised learning algorithms, [2] was able to perform better in terms of accuracy and run-time gain. A supervised learning algorithm is given pairs of data samples, with the desired output, and a functional relationship is generated. It was found that linear regression had a run-time gain of about 2K over neural networks, however, neural networks are more accurate. Overall, this paper demonstrates that two outputs of OPF may be calculated, i.e. cost, and feasibility, by approximating the problem using supervised learning algorithms.

2.3. Machine Learning to Predict Grid Faults

Paper [12] talks about something slightly different, yet still relevant to this research project. It discusses the use of machine learning algorithms to predict grid faults, such as feeder failures, asset failures, manhole events vulnerability, etc. The process diagram used for failure prediction is shown in Figure 3.

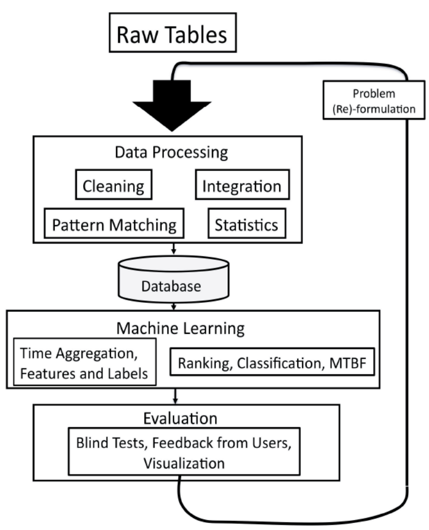

Figure 3. Process Diagram for Failure Prediction

Machine learning algorithms and analytical tools like ranking and mean time between failure (MTBF) are used to predict when an asset might fail. The data used are historical utility data, in addition to real-time data streamed from assets capable of having sensors, such as transformers. The data is processed using supervised learning algorithms to provide utilities with the ability to perform predictive maintenance, as opposed to corrective maintenance. Although [12] does not explicitly deal with power flow optimization, the methods of failure prediction using supervised machine learning can be an asset for this research project. This is because the process of failure prediction could be replicated to predict the power flow between two nodes in a power system.

3. Data

Two types of data were collected and used:

1. Time-series electricity load data from New England ISO [5]
    (a) Across the New England region (CT, MA, ME, NH, RI, VT)
    (b) Hourly granularity
    (c) Monthly load (MWh) from 03/01/2011 to 12/31/2020 for training, 01/01/2021 to 12/31/2021 for validation
    (d) Extra data was collected which was useful for data visualization: Total system load (all of New England) from 2019 to 2022 in 5 minute granularity
2. Power flow data generated from Siemens PSS/E simulations [13]
    (a) Using the IEEE 39 bus system
    (b) Customized system parameters
    (c) Different scenarios, including but not limited to, varying levels of renewables penetration, system demand, time of day, and season

These datasets satisfy the 3V's of big data:

1. Volume: The data is gathered from two sources; the ISO-NE data is over 9 years and the simulation data was over 70 scenarios
2. Velocity: Mixed velocity among the datasets; PSS/E was fast flow as the power flow computation for the model is relatively quick (even though the data was collected manually), and the load data was slower and done in batches
3. Variety: The datasets are distinct, with the load data in MWh while the simulation data contained voltage in per unit, bus number, and power flow results, i.e. system convergence/divergence. The simulation data also accounted for a wide variety of power flow scenarios.

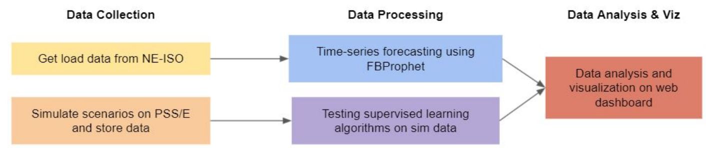

Figure 4. Overall System Architecture

4. System Overview

Figure 4 shows the proposed overall system architecture, which will be discussed in detail in the subsequent sections.

5. Methodology

5.1. Data Collection and Simulations

Firstly, electricity load data was collected from the New England ISO website using their Web Services API to automate the task [9] as seen in Figure 5. The load data that was collected through the API was the monthly load of the entire system as seen in Figure 6. There was no information on load per zone. The load was only from 2019 onward with 5 minute granularity. Therefore, more useful data, going back further in time, had to be collected manually as the API only provided access to data from 2019 onward. Load data, per zone, with hourly granularity, was collected from their publicly available web page [6] as shown in Figure 7. This load data was from 03/01/2011 present. The data was split accordingly for training and testing purposes. The data was in the form of MWh for the entire New England region.

```
isone = client_factory('ISONE', timeout_seconds=60)

for year in range(2019,2022):
    for month in range(1,13):
        last_date = calendar.monthrange(year, month)[1]
        data = isone.get_load(latest=False, start_at=f'{month}/01/{year}', end_at=f'{month}/{last_date}/{year}')
        temp_df = pd.DataFrame(data)
        temp_df.to_csv(f"{year}-{month:02d}.csv", index=False)
        print(f"{year}-{month:02d} done")
        #print(temp_df)
```

Figure 5. ISO-NE Web API Code to Scrape Data

	A	B	C	D	E	F	G
1	load_MW	SystemLoadBtmPv	NativeLoadBtmPv	timestamp	ba_name	market	freq
2	13526.008	13526.008	13267.108	2019-03-01 05:00:00+00:00	ISONE	RT5M	5m
3	13472.397	13472.397	13213.897	2019-03-01 05:05:00+00:00	ISONE	RT5M	5m
4	13423.584	13423.584	13166.184	2019-03-01 05:10:00+00:00	ISONE	RT5M	5m

Figure 6. Total Load for all of New England

Secondly, using the IEEE 39 bus model on Siemens PSS/E, shown in Figure 8, 70 different scenarios were manually run to generate the power flow datasets. The simulations were run manually because the student version of Siemens PSS/E does not allow for the incorporation of Python packages to automate processes. The data was Figure 1 shows the power flow equations for just two buses, however, for the simulations there will be 39 buses and each bus has a relationship with another bus. Therefore, each

	A	B	C	D
1	Date	Hr_End	DA_Demand	RT_Demand
2	1-Jan-17	01	997.60	1,110.97
3	1-Jan-17	02	949.80	1,064.01
4	1-Jan-17	03	920.70	1,025.41
5	1-Jan-17	04	908.20	1,020.88
6	1-Jan-17	05	892.30	1,034.40
7	1-Jan-17	06	976.20	1,052.38
8	1-Jan-17	07	1,007.40	1,089.93
9	1-Jan-17	08	1,131.60	1,152.03
10	1-Jan-17	09	1,191.30	1,229.47
11	1-Jan-17	10	1,258.50	1,280.88
12	1-Jan-17	11	1,283.00	1,308.20
13	1-Jan-17	12	1,272.10	1,315.36
14	1-Jan-17	13	1,257.70	1,311.75
15	1-Jan-17	14	1,249.30	1,295.21
16	1-Jan-17	15	1,261.70	1,287.62
17	1-Jan-17	16	1,288.90	1,308.14
18	1-Jan-17	17	1,477.30	1,423.74
19	1-Jan-17	18	1,509.10	1,518.59
20	1-Jan-17	19	1,457.70	1,486.47
21	1-Jan-17	20	1,434.90	1,431.33
22	1-Jan-17	21	1,375.70	1,352.08
23	1-Jan-17	22	1,255.90	1,265.31
24	1-Jan-17	23	1,150.80	1,180.97
25	1-Jan-17	24	1,045.10	1,122.53

Figure 7. Monthly Load for Zone of Maine

simulation calculates 1521 power flow relationships, with an additional 1521 relationships to take into account the bilateral flow of power in a power system, for a total of 3042 power flow relationships that have to be optimized.

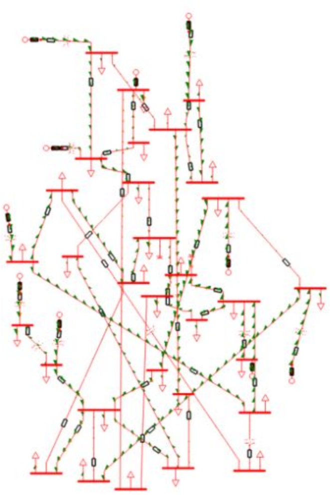

Figure 8. IEEE 39 Bus Model on PSS/E

The simulation scenarios included the base case, followed by implementing various levels of solar penetration according to the landscape of New England. An example scenario that was run used the existing load. It simulated a peak summer day in New England with 15 hours of sunlight. The existing renewable energy generation capacity was simulated [7]. Wind, solar, and storage constitute 4%, 2%, and 2% of existing renewable energy generation, respectively. Taking into account the capacity factor of each generation type, the effective energy output of wind, solar, and storage were 0.12%, 0.0708%, and 0.075%, respectively. Figure 9 shows the bus voltages for the first 10 buses (out of 39) for the entire model after the simulation. Figure 10 shows that the system was able to converge for the given simulation. And finally, Figure 11 shows the power flow relationships for bus 1.

Bus Number	Base kV	Voltage (pu)	Angle (deg)	Normal Vmax (pu)	Normal Vmin (pu)	Emergency Vmax (pu)	Emergency Vmin (pu)
1	1.0	1.0789	-8.13	1.1000	0.9000	1.1000	0.9000
2	1.0	1.0968	-5.92	1.1000	0.9000	1.1000	0.9000
3	1.0	1.0809	-8.47	1.1000	0.9000	1.1000	0.9000
4	1.0	1.0630	-9.13	1.1000	0.9000	1.1000	0.9000
5	1.0	1.0635	-8.02	1.1000	0.9000	1.1000	0.9000
6	1.0	1.0614	-7.36	1.1000	0.9000	1.1000	0.9000
7	1.0	1.0500	-9.35	1.1000	0.9000	1.1000	0.9000
8	1.0	1.0488	-9.81	1.1000	0.9000	1.1000	0.9000
9	1.0	1.0622	-9.67	1.1000	0.9000	1.1000	0.9000
10	1.0	1.0666	-5.20	1.1000	0.9000	1.1000	0.9000

Figure 9. Bus Voltages for the Model

```
Reached tolerance in 4 iterations

Largest mismatch:      -0.10 MW      -0.00 Mvar       0.10 MVA at bus 29 [29          1.0000]
System total absolute mismatch:                       0.75 MVA

SWING BUS SUMMARY:
  BUS#-SCT X-- NAME --X BASKV      PGEN     PMAX    PMIN      QGEN     QMAX     QMIN
    31     31            1.0000     568.7   9999.9    0.0      54.2*    0.0      0.0
```

Figure 10. Convergence Result for the Simulation Scenario

```
BUS      1      1          1.0000 CKT     MW      MVAR      MVA    % 1.0793PU
                                                                     1.0793KV

  TO     2      2          1.0000  1   -118.2    -77.6    141.4

  TO    39     39          1.0000  1    118.2     77.6    141.4
```

Figure 11. Power Flow Relationships for Bus 1

Another simulation example similarly used the existing load. However, this time it was a peak summer night which consisted of 9 hours. The renewable energy generation constituents remained the same. However, due to it being nighttime, the capacity factor changed. Therefore, the effective energy generation of the three sources dropped to 0.01% wind, 0% solar, 0.075% storage. As noted, the effective energy generated from storage remains the same as it is not affected by a lack of sunlight. Various other scenarios were run to generate power flow data to be used for training a model to be able to predict the power flow results of this system. The initial idea was to use PyPSA [11] and/or psspy [10] to automate the simulations, however, the student version of Siemens PSS/E used in this project does not support those packages. Therefore, the simulations were done manually.

5.2. Energy Demand Forecasting using FBProphet

After getting the electricity load dataset, the time-series forecasting was done. The `FBProphet` library was be used for the forecasting. FBProphet [15] does the forecasting by either additive or multiplicative model, and adjusts trends based on seasonalities of all types (regular daily, weekly, yearly, and also irregular holidays). This way, the library is robust to trends shifts, and outliers.

`pandas-gbq` library was used for querying load data from storage. Once the load data was loaded and formatted in the required format for FBProphet a prediction model was trained for each state. Figure 12 shows the model being trained for the state of Maine. FBProphet allows for the consideration of holidays based on countries. Naturally, during holidays in the United States the energy usage will change, especially during large public events like New Year's eve or July 4th. The model is able to account for these seasonalities and take it into the prediction.

```
model = Prophet(
        changepoint_prior_scale=0.5,
        seasonality_mode='multiplicative',
        interval_width=0.95,
    )
model.add_country_holidays(country_name='US')
```

Figure 12. FBProphet Model Training for Maine

Once a prediction model is trained for each state in New England a prediction can be made in any future time scale. Figure 13 shows a prediction being made for one year into the future. The model was trained with data up to 2021. Figure 14 shows the results of the future prediction where yhat is the prediction for energy demand (MWh) on that particular day, and yhat_lower and yhat_upper are the lower and upper limits on the prediction, respectively. Figure 15 shows the plotted forecast_pd with their upper and lower limits.

```
future_pd = model.make_future_dataframe(
        periods=365,
        freq='1H',
        include_history=True
    )

# make predictions
forecast_pd = model.predict(future_pd)
```

Figure 13. Loading One Year Ahead Prediction for Maine

	ds	yhat	yhat_lower	yhat_upper
97536	2022-02-16 00:00:00	1222.310143	1054.757560	1401.791215
97537	2022-02-16 01:00:00	1181.339685	1006.014964	1343.711103
97538	2022-02-16 02:00:00	1160.001157	997.737491	1337.266448
97539	2022-02-16 03:00:00	1165.047049	983.699111	1348.152193
97540	2022-02-16 04:00:00	1207.842258	1041.070343	1384.551975

Figure 14. One Year Ahead Demand Prediction for Main

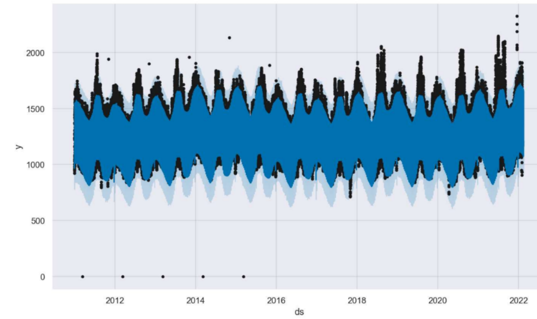

Figure 15. Forecast for Maine

An additional capability of the FBProphet library is the visualization metrics. For example, trend graphs can be generated based on daily, weekly, or monthly intervals for each model. As the demand is forecasted these trend graphs can prove useful to quickly analyze how demand is changing based on different time intervals. The weekly and yearly trend graphs for the state of Maine are shown in figures and 16 and 17.

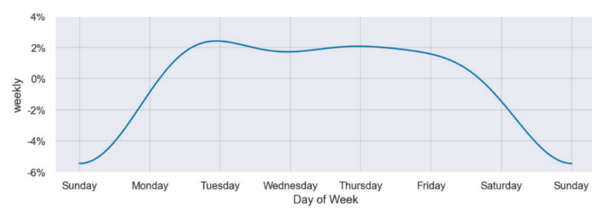

Figure 16. Weekly Trend for Maine

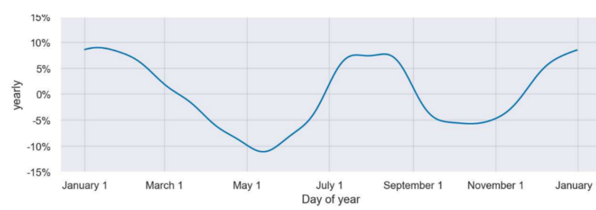

Figure 17. Yearly Trend for Maine

The model is able to accurately predict the trend for the subsequent years because Maine is actually a summer peak load. That means Maine's energy demand peaks during the summer months as more people use air conditioning to cool their homes. One might think that Maine being up north could lead to a winter peak as people use heaters to warm their homes. However, while more people may use heaters to warm their homes during the longer winter months than ACs to cool their homes during the shorter summer months, heating systems in Maine are largely powered by natural gas. Therefore, unless they are electric heaters, which are few and far in between, the use of heaters is not reflected in the energy usage as much as the use of ACs is. The weekly trend also reflects natural energy usage trends accurately because the load is high from Monday through Friday which are working days.

The above process was repeated to forecast energy demand 10 years into the future. The reason why 10 years was the chosen time scale is because the model might be inaccurate for anything longer, as energy demand can fluctuate greatly depending on climate conditions, infrastructural projects, etc. Figure 18 shows a sample of the 10-year ahead forecast for the state of Maine, until the year 2032.

ds	yhat_lower	yhat_upper	yhat
1/25/2032 23:00	-10369.62409	15405.18696	1681.183389
1/26/2032 23:00	-10903.55721	16112.63375	1743.043234
1/27/2032 23:00	-10765.81479	15940.92842	1729.224355
1/28/2032 23:00	-10732.91796	16009.94647	1734.063772
1/29/2032 23:00	-10667.75884	15893.82721	1724.319123

Figure 18. Sample of 10-Year Ahead Forecast for Maine

5.3. Power Flow Prediction Using Supervised Learning Algorithms

For the data from the simulations, multiple supervised learning algorithms, such as linear regression and kNN, and Decision Tree Regressor were used to predict the power flow under different scenarios. The input data was in the format shown in Figure 19. The bus values range from 1 to 39. Nominal voltage values are from 0.9 to 1.1, with abnormal voltage values falling outside this range. There was a 50/50 mix of nominal and abnormal data to be able to train the model appropriately. Season was separated into 1 and 0 with 1 meaning summer and 0 meaning winter. Time was also separated into 1 and 0 meaning day and night, respectively. This was done because, as explained in a previous section, the effective power output of solar and wind energy varies depending on season, and time of day.

Bus	Voltage	Season	Time	Demand	Renewables	Output
1	1.0796	1	1	100	8	1
2	1.0988	1	1	100	8	1
3	1.0841	1	1	100	8	1
4	1.065	1	1	100	8	1
5	1.0643	1	1	100	8	1
6	1.062	1	1	100	8	1
7	1.0506	1	1	100	8	1
8	1.0493	1	1	100	8	1
9	1.0623	1	1	100	8	1

Figure 19. Input Data for Power Flow Prediction

Additionally, the Demand column has values of 100% to 200%. 200% demand was chosen as the upper limit, even though the energy demand predictions from FBProphet did not indicate a 100% increase in demand, in order to intentionally break the system to generate the abnormal data. The Renewables column has values from 8% to 50% because these seemed like appropriate values for this study. New England currently generates 8% of its energy from renewable sources hence why that value was chosen. 50% was chosen because this is a medium- to long-term study and renewable penetration above 50% is likely more than 30 years away, therefore, it is impractical for this study. Output was either 1, for system convergence, or 0 for system divergence.

Firstly, multi-feature linear regression was used where the input columns were Season, Time, Demand, and Renewables. The output was Voltage and Output. The model was able to predict the system convergence, i.e. output along with a value for voltage for each bus. If voltage was also considered as an input feature then the model would likely be more accurate, however, our goal is to also predict the voltage values of each bus. The model had a root mean squared error (RMSE) of 0.437 and an R-squared value of 0.50. Both of these values are acceptable and prove that linear regression is a viable model to use.

Next, k-Nearest Neighbors (kNN) was used to perform the same functions. The same input features were used to predict the same outputs. The kNN model was 100% accurate which indicated that the model was over-fitting the data. There was relatively limited simulation data, due to the fact that the 70 simulations had to be run manually which was time-consuming. However, future work can incorporate other model kinds of model research such as decision trees.

Lastly, Decision Tree Regressor was tested as shown in Figure 20. The same inputs were used and the model performed well, with an R-squared value of 0.72 and an RMSE of 0.015. Naturally, due to the better performance, the Decision Tree Regressor model was used in the dashboard visualization. The output of the model is an average system voltage in per unit, and power flow results, i.e. system convergence/divergence.

```
clf = DecisionTreeRegressor(random_state=0)
```

```
#Training the decision tree classifier.
clf.fit(X_train, y_train)

y_pred =  clf.predict(X_test)
X_test
```

Figure 20. Testing the Decision Tree Regressor

5.4. Data Analysis and Visualization

The results of the FBProphet time-series forecasting determined an estimate of the future energy demand of the New England area. Since FBProphet takes into account seasonalities and outliers it is a reliable estimate for the purposes of this project. The graphical visualizations further cemented its accuracy.

The results of the simulation indicate whether the PSS/E model can converge, i.e. solve for power flow under those constraints. The Decision Tree Regressor model was able to identify an optimal system value for voltage under user-specified conditions and overall power flow results. This was also the basis for the web dashboard visualization.

The narrative of the visualization is the following: *Can the New England system support itself, i.e. converge to optimal power flow, for different generation and usage scenarios, and meet the estimated energy demand of the future?* It was built using Flask for the back end and HTML/CSS and Plotly for the front end.

Figure 21 shows the Home page of the web dashboard. The dashboard allows for the user to have 4 inputs: Season (1/0 = Summer/Winter), Time of Day (1/0 = Day/Night), Demand (0% to 200%), and Renewables (0% to however much the user wishes). The dashboard takes the user inputs and generates an output in the form of [System Voltage, Power Flow Result] with the power flow result being in the form of 1/0 = convergence/divergence.

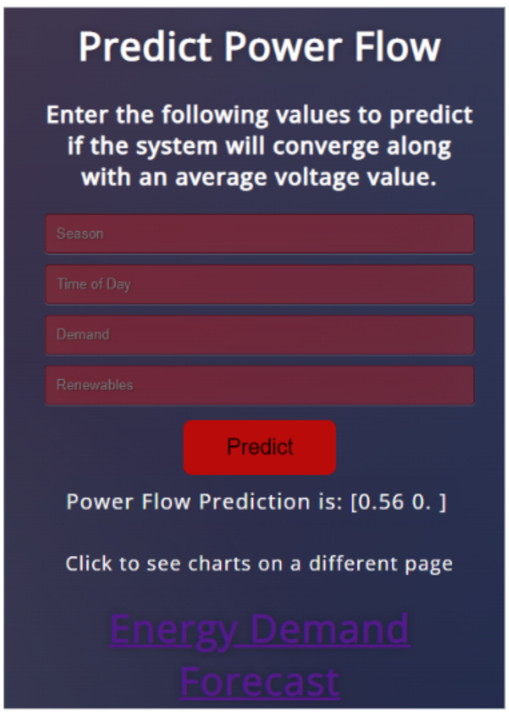

Figure 21. Home Page of Web Dashboard

The link below the power flow result titled "Energy Demand Forecast" directs the user to a page with a graphical visualization of the energy demand forecast of all zones in New England. This can be seen in Figure 22. The image in the webpage shows the color-coordinated forecasted energy demand for all zones in New England up until the year 2032.

The user is able to adjust the time scale and roll over each trend to view the specific data point. This is intended to provide the user with context to view their inputs in. For example, if a user wants to check if the system will converge

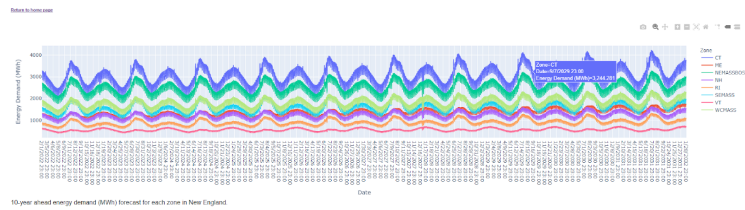

Figure 22. Energy Demand Forecast Page of Web Dashboard

with 40% renewables and 110% demand in the year 2030, they need only enter the inputs in the home page, check the power flow results, and then navigate to the forecast page to check the demand in that year. Click here to see a video of the web dashboard demo.

6. Future Work

While a single ML model may not be able to accurately predict the power flow of a real-world complex system with constantly fluctuating values of voltage and reactance, this proof of concept can be worked upon to come up with more powerful models that can rival modern methods of computing power flow.

Additional work can be done to refine the ML model, such as generating more simulation training data with increased volume and complexity to encompass more real-world scenarios. The ML model itself has to be more accurate in order to justify use in a business case. This is because power flow has to be 100% accurate as people's lives and livelihoods depend on energy.

Another improvement could be to generate voltage values for each bus, however, since the goal of the project is to make power flow faster, this could possibly impede that. Finally, currently users on the web dashboard have to manually correlate their inputs with the demand forecast. Finding a way to automate the inputs would be an appropriate next step, for example, by adding an additional user input of Year, and an additional output that zooms the forecast graph on that year.

7. Discussion and Conclusion

As populations rise so will energy needs. This will lead to larger and more complex power systems that have to serve the need of the people, while also accommodating for intermittent renewable energy systems. Consequently, the power flow of these systems must be computed quickly and accurately. This will take immense computational power, which is why if there are methods to hasten these computations, they must be vigorously explored.

The system mentioned in this paper does just that, i.e. lowers the computation power and time used to solve for optimal power flow via the use of machine learning. Using a Decision Tree Regressor model trained with data generated from simulations using real-world constraints, a prediction of system convergence can be made to impact real-time business decisions. This model avoids having to deal with computing the power flow at each bus, but instead is able to predict the power flow of the entire system with reasonable accuracy.

This system would be especially useful for certain sectors of industry. Firstly, technical users, such as Power Systems Engineers could use this system to perform quick power flow predictions to determine system health under certain conditions, such as over-/under-load scenarios. Secondly, non-technical users could use this system for the following reasons: lawmakers could use the system to strengthen public policy decisions. For example, when determining whether or not to approve a new solar project, they could use this method to see how the system would perform with increased renewable integration. Additionally, financial institutions could use this method to determine financial viability when deciding whether or not to fund renewable energy projects.

Overall, the system described in this paper not only optimizes power flow, but also opens up this field to users who may not have technical expertise. This is valuable, especially because the power system of the future will require cooperation, not only from engineers, but also from lawmakers, financial enterprises, educators, etc. Therefore, enabling increased cooperation amongst everyone, will allow us to prepare for the power system of the future.

References

[1] Mary B. Cain and Richard P. O'Neill. History of optimal power flow and formulations. 2012. 1

[2] Raphael Canyasse, Gal Dalal, and Shie Mannor. Supervised learning for optimal power flow as a real-time proxy, 2016. 2

[3] Neel Guha, Zhecheng Wang, Matt Wytock, and Arun Majumdar. Machine learning for ac optimal power flow, 2019. 1

[4] Fouad Hasan, Amin Kargarian, and Ali Mohammadi. A survey on applications of machine learning for optimal power flow. In *2020 IEEE Texas Power and Energy Conference (TPEC)*, pages 1–6, 2020. 2

[5] New England ISO. Energy, load, and demand reports. https://www.iso-ne.com/isoexpress/web/reports/load-and-demand, Feb. 2022. 3

[6] ISO-NE. https://www.iso-ne.com/isoexpress/web/reports/load-and-demand. https://www.iso-ne.com/isoexpress/web/reports/load-and-demand, Mar. 2022. 3

[7] ISO-NE. Iso-ne forecast. https://www.iso-ne.com/static-assets/documents/2021/03/new_england_power_grid_regional_profile.pdf, Feb. 2022. 4

[8] Sidhant Misra, Line Roald, and Yeesian Ng. Learning for constrained optimization: Identifying optimal active constraint sets, 2019. 2

[9] NEISO. Iso new england web services api v1.1. https://webservices.iso-ne.com/docs/v1.1/, Feb. 2022. 3

[10] NREL. Pypsse. https://www.nrel.gov/grid/pypsse.html, Feb. 2022. 4

[11] PyPSA. Pypsa: Python for power system analysis. https://pypsa.readthedocs.io/en/latest/index.html, Feb. 2022. 4

[12] Cynthia Rudin, David Waltz, Roger N. Anderson, Albert Boulanger, Ansaf Salleb-Aouissi, Maggie Chow, Haimonti Dutta, Philip N. Gross, Bert Huang, Steve Ierome, Delfina F. Isaac, Arthur Kressner, Rebecca J. Passonneau, Axinia Radeva, and Leon Wu. Machine learning for the new york city power grid. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(2):328–345, 2012. 2, 3

[13] Siemens. Pss®e – high-performance transmission planning and analysis software. https://new.siemens.com/global/en/products/energy/energy-automation-and-smart-grid/pss-software/pss-e.html, Feb. 2022. 3

[14] Yixuan Sun, Xiaoyuan Fan, Qiuhua Huang, Xinya Li, Renke Huang, Tianzhixi Yin, and Guang Lin. Local feature sufficiency exploration for predicting security-constrained generation dispatch in multi-area power systems. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1283–1289, 2018. 2

[15] Sean J Taylor and Benjamin Letham. Forecasting at scale. *The American Statistician*, 72(1):37–45, 2018. 5

[16] Wikipedia. Power-flow study. https://en.wikipedia.org/wiki/Power-flow_study, May 2013. 1