# Robust Training for AC-OPF (Student Abstract)

**Fuat Can Beylunioğlu**[1], **Mehrdad Pirnia**[1], **P. Robert Duimering**[1], **Vijay Ganesh**[2]

[1] Management Sciences, University of Waterloo
[2] Computer Science, University of Waterloo
{fcbeylun, mpirnia, rduimering, vijay.ganesh}@uwaterloo.ca

## Abstract

Electricity network operators use computationally demanding mathematical models to optimize AC power flow (AC-OPF). Recent work applies neural networks (NN) rather than optimization methods to estimate locally optimal solutions. However, NN training data is costly and current models cannot guarantee optimal or feasible solutions. This study proposes a robust NN training approach, which starts with a small amount of seed training data and uses iterative feedback to generate additional data in regions where the model makes poor predictions. The method is applied to non-linear univariate and multivariate test functions, and an IEEE 6-bus AC-OPF system. Results suggest robust training can achieve NN prediction performance similar to, or better than, regular NN training, while using significantly less data.

## Introduction

The AC-OPF problem consists of finding optimal system variables, such as real and reactive power generation and set points in generator buses, to satisfy user demand and minimize costs. System operators update optimization models frequently to adapt to electricity supply and demand uncertainty. Failure to find optimal and feasible solutions may result in increased costs, under-utilization of renewables, or even system failure. Quasi Newton Raphson solvers are traditionally used to find optimal decision variables of the Lagrangian. The difficulty is that computational cost increases exponentially with system size due to the Jacobian matrix inversion procedure. The problem is also nonlinear, so finding the global minimum is not guaranteed. This led researchers to train neural networks (NN) on large datasets of matched input power demands and outputs representing the optimal supply from each generator (Huang and Wang 2022). Although current NN models provide solutions quickly, they rely on large training sets reflecting the full range of potential system behavior. However, this is costly due to the need to run many traditional optimizations to generate sufficient data and may not provide accurate solutions due to out-of-range input variables (Lee et al. 2018).

This study applies a robust NN training approach (Scott, Panju, and Ganesh 2020; Bengio et al. 2021) to improve the optimality and feasibility of predictions for the AC-OPF problem, while using substantially less training data than typical NN approaches. The approach uses iterative feedback to generate targeted training data to improve NN performance for cases where model predictions are poor or violate feasibility constraints. Training starts with a seed sample for validation and a randomly initialized network. At each iteration the model predicts on a previously generated validation set and prediction errors are assessed. Then poorly predicted validation examples are selected and populated via small perturbations to create new training data in the neighborhood of the poor predictions, to be used in the next iteration. Training continues until there are no further validation losses, or a pre-specified number of iterations is reached.

The method is first applied to two problems representing highly non-linear univariate and multivariate algebraic functions, to explore the dynamics, strengths and weaknesses of the procedure and to choose appropriate hyperparameters. The method is then applied to a 6-bus AC-OPF NN model to compare the relative performance of robust versus traditional NN training.

## Methodology

The proposed method is illustrated in Figure 1. To reduce data demands, a randomly sampled portion of the original training set $T$ is used to create the set $T^0$, which serves as initial set $T^i$ for the first iteration. Then, a random subset of $T^i$ is used to create the initial robust training set $T^r$, which is updated through subsequent iterations:

- Initially, the NN is started with random weights.
- The resulting NN is tested for error on each observation in $T^i$. If the error is greater than the specified threshold, then the observation is classified as a poor prediction. Otherwise, the data point is ignored.
- A portion of the resulting poor prediction set is randomly selected to create set $t$.
- Each input observation in $t$ is perturbed to create multiple points, and their corresponding target values are calculated via a solver, creating set $t'$, the size of which is a multiple of $t$, depending on the number of perturbations.
- The new set $t'$ is added to the set $T^r$, which is used to re-train the NN.
- Set $t'$ is also added to the original set $T^i$, where its subset is part of the robust set $T^r$.

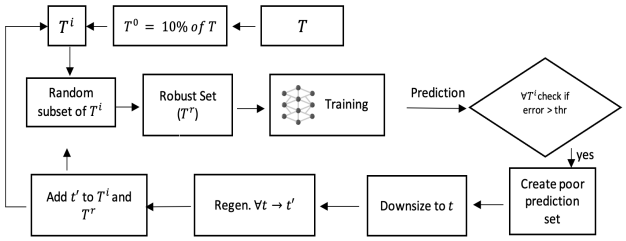


Figure 1: Robust Training

Set $T^i$ expands after each iteration, as each new set $t'$ is generated and added. The size of set $T^r$ stays constant, although its components keep changing. Data were generated two ways: (i) perturbing inputs randomly; (ii) differentiating validation loss with respect to the inputs to find the direction that increases the loss and identify challenging examples to improve performance.

## Numerical Examples

The robust training method was implemented on highly nonlinear function (HNL), $f(\mathbf{x}) = \sin(\mathbf{x}) + \cos(\mathbf{x}) + \mathbf{x}/5$ with $\mathbf{x} = [x]$ for the univariate case and $\mathbf{x} = [x_1, x_2]$ for the multivariate case, and also deployed on a 6-bus AC-OPF problem. For both traditional and robust training a deep NN architecture was used, with two hidden layers of 300 neurons, Tanh activation function between layers, and a batch normalization layer after the input layer to normalize the wide range of input values. Additionally for the AC-OPF case, a final sigmoid layer is added to the output layer.

For the HNL functions, 10000 datapoints were generated by randomly drawing inputs $\mathbf{x}_j \sim \mathcal{U}(0, 20)$, and calculating the output as $y = f(\mathbf{x})$. For the AC-OPF problem, a base input demand series was populated by multiplying with a constant drawn from $\mathcal{U}(0.6, 1.4)$ and running the PandaPower AC-OPF solver to generate 20000 datapoints. Inputs included real and reactive power demand $P_D$ and $Q_D$ at each demand bus, and target values $y$ consisted of real, reactive, and voltage magnitude $P_G, Q_G, V_G$ for each generator bus in the output layer. Unlike the HNL examples, optimal target values were not calculated for each new perturbed input data point generated during the robust training iterations due to time limitations. Instead, a set of 16000 examples were sampled prior to the analysis; then during each robust training iteration, target values for new inputs were selected from this set by identifying examples with the closest euclidean distance to the newly generated inputs.

The data were randomly split using a 70/10/20 ratio for train, validation and test sets, respectively. For regular training, the network was trained for 100 epochs using Adam optimizer with learning rate = $0.001$ and batch size = $512$. For robust training, cut-off threshold $= 0.01$, maximum iteration = $10$ for $70$ training epochs. At each epoch, data points, yielding poor predictions were selected and downsized randomly to $150$ examples. For each example, $2$ new data points were generated by perturbing the inputs with a uniform random multiplier (Random Samp) drawn from $\mathcal{U}(0.80, 1.20)$. For the HNL functions, new data were also generated using directional sampling (Dir Samp) by adding noise in the direction that maximizes the the loss. The initial robust sample ($T^0$) was selected randomly ([Rand]) for the AC-OPF case; and either randomly or using evenly spaced inputs ([Eq]), e.g. $[0, 0.001, 0.002, \ldots 20]$ for the HNL cases.

| Training | Univar | Multivar | 6-Bus |
|---|---|---|---|
| Regular (large set) | 0.0101 | 0.0896 | 1.7030 |
| Regular (small set) | 0.0165 | 0.0975 | 2.1582 |
| Random Samp [Rand] | 0.0110 | 0.0449** | 1.1963** |
| Random Samp [Eq] | 0.0054** | 0.2092 | |
| Dir Samp [Rand] | 0.0066** | 0.1560 | |
| Dir Samp [Eq] | 0.0090** | 0.1800 | |

Table 1: MSE for HNL and 6-Bus examples ($^{**}$ $p \leq 0.01$)

To show the effectiveness of robust training, regular training was repeated with smaller samples, equal to those used in robust training (small set). NN training was repeated 300 times for each method and average mean squared errors (MSE) are reported in Table 1. Two-sided Welch's t-tests were used to compare robust and regular training MSE performance on 300 sample outcomes. The robust models required less intial training data than regular training but resulted in similar or lower MSE and also took less time to train. For example, the robust HNL multivariate (Random Samp [RAND]) model required 19% less training time than regular training (avg. 7.0 vs. 8.6s per training).

## Conclusion and Future Work

The univariate, multivariate, and 6-bus AC-OPF results suggest that robust training can improve the performance of NN-aided AC-OPF solvers, achieving similar or better predictions, while using significantly less training data than traditional methods. The method offers potential efficiency gains relative to regular training, because a feedback mechanism is used to generate new training data only for regions of the input domain where predictions are poor. Further research is needed to investigate the performance of robust training on larger, more complex AC-OPF problems, and to better understand the effects of various hyper-parameters on performance.

## References

Bengio, E.; Jain, M.; Korablyov, M.; Precup, D.; and Bengio, Y. 2021. Flow network based generative models for non-iterative diverse candidate generation. *Adv. in Neural Information Processing Systems*, 34: 27381–27394.

Huang, B.; and Wang, J. 2022. Applications of Physics-Informed Neural Networks in Power Systems-A Review. *IEEE Trans. on Power Systems*. Forthcoming.

Lee, K.; Lee, H.; Lee, K.; and Shin, J. 2018. Training Confidence-calibrated Classifiers for Detecting Out-of-Distribution Samples. In *Proc. ICLR*.

Scott, J.; Panju, M.; and Ganesh, V. 2020. Lgml: Logic guided machine learning. *Proc. AAAI*, 34(10): 13909–13910.