Addition to the paper: End-to-End Feasible Optimization Proxies for Large-Scale Economic Dispatch

Ilay Menachem

February 22, 2026

1 Introduction

The paper proposes an End-to-End Learning and Repair (E2ELR) architecture for training optimization proxies for economic dispatch problems. That is, at the end of the network, there are repair layers that ensure power balance and reserve. As stated in the paper, if there is a feasible solution, the repair layers will output a feasible solution, and if there is no feasible solution, the repair layers will minimize in some sense the violation of the constraints. In this work, we merge the ED problem with the DC-OPF problem, and we propose to add a layer that will ensure the satisfaction of the DC power flow equations.

2 Problem Statement

Here is a problem statement that merges the problem statement in the paper with the DC-OPF problem statement.

$$\min_{\boldsymbol{p},\boldsymbol{r},\theta} c(\boldsymbol{p}) + penalty(\boldsymbol{p},\boldsymbol{r},\theta) \tag{1}$$

subject to

1. $e^T\boldsymbol{p} = e^T\boldsymbol{d}$ (global power balance)

2. $e^T\boldsymbol{r} \geq R$ (minimum reserve requirement)
3. $\boldsymbol{p} + \boldsymbol{r} \leq \overline{\boldsymbol{p}}$ (generator output limits)
4. $0 \leq \boldsymbol{p} \leq \overline{\boldsymbol{p}}$ (generator output limits)
5. $0 \leq \boldsymbol{r} \leq \overline{\boldsymbol{r}}$ (generator reserve limits)
6. $\forall i, j \; \frac{\theta_i - \theta_j}{x_{ij}} \leq \overline{P}_{ij}$ (line flow limits)
7. $\boldsymbol{B}\theta = \boldsymbol{p} - \boldsymbol{d}$ (DC power flow equations)

and for models which don't guarantee the feasibility of solution, we add penalty terms to the cost function.

$$penalty_{pb}(\hat{\boldsymbol{p}}) = M_{pb} \cdot |e^T\hat{\boldsymbol{p}} - e^T\boldsymbol{d}|$$

$$penalty_{res}(\boldsymbol{r}) = M_{res} \cdot Relu(R - e^T\boldsymbol{r})$$

$$penalty_{th}(\hat{\boldsymbol{p}}, \theta) = M_{th} \cdot \sum_{i,j} Relu(\frac{\theta_i - \theta_j}{x_{ij}} - \overline{P}_{ij})$$

$$penalty_{dc}(\hat{\boldsymbol{p}}, \theta) = M_{dc} \cdot \|\boldsymbol{B}\theta - (\hat{\boldsymbol{p}} - \boldsymbol{d})\|_1$$

$$penalty(\hat{\boldsymbol{p}}, \theta) = penalty_{pb}(\hat{\boldsymbol{p}}) + penalty_{res}(\hat{\boldsymbol{r}}) + penalty_{th}(\hat{\boldsymbol{p}}, \theta) + penalty_{dc}(\hat{\boldsymbol{p}}, \theta)$$

As seen in the paper, conditions 2, 3, and 5 are equivalent to the condition $R \leq e^T \min(\overline{\boldsymbol{r}}, \overline{\boldsymbol{p}} - \boldsymbol{p})$, and removing $\boldsymbol{r}$ from the optimization variables.

Thus the problem can be rewritten as:

$$\min_{\boldsymbol{p},\boldsymbol{\theta}} c(\boldsymbol{p}) + penalty(\boldsymbol{p}, \theta) \tag{2}$$

subject to

1. $0 \leq \boldsymbol{p} \leq \overline{\boldsymbol{p}}$ (generator output limits)
2. $e^T\boldsymbol{p} = e^T\boldsymbol{d}$ (global power balance)
3. $R \leq e^T \min(\overline{\boldsymbol{r}}, \overline{\boldsymbol{p}} - \boldsymbol{p})$ (minimum reserve requirement)
4. $\boldsymbol{B}\theta = \boldsymbol{p} - \boldsymbol{d}$ (DC power flow equations)
5. $\forall i, j \; \frac{\theta_i - \theta_j}{x_{ij}} \leq \overline{P}_{ij}$ (line flow limits)

where the penalty of the reserve constraint is now:

$$penalty_{res}(\hat{\boldsymbol{p}}) = M_{res} \cdot Relu(R - e^T \min(\overline{\boldsymbol{r}}, \overline{\boldsymbol{p}} - \hat{\boldsymbol{p}}))$$

One should note that the penalty terms in the paper are all much greater than the cost of producing energy, that is to say the main concern of the neural network training process is to satisfy the constraints, and not to minimize the cost function. thus, it is easy to see why adding repair layers would yield a much more effective network.

3 Proposed Methods

All the networks in this work will also output voltage angles θ.

In this work, in addition to the repair layers introduced in E2ELR, we propose to add a theta repair layer for the equality constraint $\boldsymbol{B}\theta = \boldsymbol{p} - \boldsymbol{d}$. This repair layer will find the θ^* which satisfies the DC power flow equation, and is closest to the θ the neural network outputs. Finding θ^* can be viewed as the following optimization problem:

$$\min_{\theta'} \|\theta' - \theta\|_2^2 \quad \text{s.t.} \quad \boldsymbol{B}\theta' = \boldsymbol{p} - \boldsymbol{d} \tag{3}$$

The solution to this problem is the projection of θ onto the affine subspace $\{\theta' : \boldsymbol{B}\theta' = \boldsymbol{p} - \boldsymbol{d}\}$, which is given by:

$$\theta^* = \theta + \boldsymbol{B}^\dagger(\boldsymbol{p} - \boldsymbol{d} - \boldsymbol{B}\theta) \tag{4}$$

where $\boldsymbol{B}^\dagger$ is the Moore-Penrose pseudo-inverse of $\boldsymbol{B}$. $\boldsymbol{B}$ is a sparse matrix, thus the pseudo-inverse can be computed efficiently and matrix-vector multiplication with $\boldsymbol{B}^\dagger$ can also be computed efficiently; every operation in equation 4 can be implemented efficiently.

4 Experiments

4.1 Experimental Setup

The experimental setup is the same as in the paper, except $\lambda = 1.0$, and for the hyperparameter tuning. The hyperparameter tuning is done using Optuna's hyperparameter optimization with 25 trials, and ASHAScheduler for early stopping. Additionally, due to the constraints added to the problem, we add the parameter $M_{dc} = 3500\frac{\$}{MW}$ for the penalty term of the DC power flow equations, which is the same as the power balance parameter M_{pb}. The datasets which were tested for are ieee300, and pegase 1354.

4.2 Results

The hyperparameter tuning results are shown in Tables 1 and 2.

Table 1: Best hyperparameters found – ieee300 dataset.

Hyperparameter	**DNN**	**E2ELR**	**E2ELRDC**
Layers	2	2	2
Hidden dim	512	256	128
Learning rate	1.135×10^{-4}	8.566×10^{-4}	1.026×10^{-4}
Batch size	128	256	128

Table 2: Best hyperparameters found – pegase1354 dataset.

Hyperparameter	**DNN**	**E2ELR**	**E2ELRDC**
Layers	4	3	2
Hidden dim	512	128	128
Learning rate	1.746×10^{-4}	7.141×10^{-4}	5.275×10^{-3}
Batch size	64	128	128

It is noteworthy that the more constrained the output of the network is, the fewer layers and smaller hidden dimension that is found to be optimal; this indicates that the extra parameters in the network are used to learn how to satisfy the constraints. For the pegase1354 dataset, the more constrained the output of the network is, the higher learning rate that is found to be optimal. The relation could be because the more constrained the output of the network is, the faster the network can learn, but this fact is not observed in the ieee300 dataset and thus it is not clear if it is a real relation or just a coincidence.

Training the networks using the hyperparameters shown above, we get the following results for the test set shown in Tables 3 and 4.

Table 3: Test set results – ieee300 dataset.

Model	**Gap (%)**	**SGM Gap (%)**	**PB Viol. (pu)**	**DC Viol. (pu)**
DNN	603.86	595.75	5.50×10^{-1}	7.527
E2ELR	736.70	730.52	6.06×10^{-6}	10.043
E2ELRDC	**21.49**	**19.94**	7.14×10^{-6}	**0.182**

Table 4: Test set results – pegase1354 dataset.

Model	**Gap (%)**	**SGM Gap (%)**	**PB Viol. (pu)**	**DC Viol. (pu)**
DNN	2748.71	2692.70	2.07×10^{1}	62.747
E2ELR	2563.26	2551.00	2.08×10^{-5}	77.578
E2ELRDC	**362.46**	**348.80**	1.98×10^{-5}	**9.848**

These results show that adding the theta repair layer significantly reduces the DC power flow violation, and also significantly reduces the optimality gap. although the addition of the theta repair layer does help it is clear that the results still leave a lot to desire, and there is still a significant gap between the E2ELRDC model and the optimal solution.

5 Conclusion

Although the idea proposed in the previous paper for using closed form repair layers is not easily extendable to new constraints. by understanding the structure of the constraints or adding constraints with new variables, it is possible to design additional and efficient repair layers, yet this will not always yield great results out of the box.