An Unsupervised Neural Network Approach for Solving the Optimal Power Flow Problem

Alexander Svensson Marcial[a] and Magnus Perninge[b]
Department of Physics and Electrical Engineering, Linnaeus University, Växjö, Sweden

Keywords: AC-OPF, Unsupervised Learning, Deep Learning.

Abstract: Optimal Power Flow is a central tool for power system operation and planning. Given the substantial rise in intermittent power and shorter time windows in electricity markets, there's a need for fast and efficient solutions to the Optimal Power Flow problem. With this in consideration, this paper propose an unsupervised deep learning approach to approximate the optimal solution of Optimal Power Flow problems. Once trained, deep learning models benefit from being several orders of magnitude faster during inference compared to conventional non-linear solvers.

1 INTRODUCTION

Optimal power flow (OPF) is a fundamental tool used in power system operation and planning. Since its initial formulation in the 1960s (Carpentier, 1962) OPF has been instrumental in numerous applications, encompassing economic dispatch, securing reactive power reserves for voltage stability (Capitanescu, 2011), generation and transmission expansion planning (Skolfield and Escobedo, 2022). Furthermore, it serves as a foundational tool for market clearing in deregulated electricity markets.

In general, OPF is a non-convex optimization problem that classifies as NP-hard (Lavaei and Low, 2011). Its objective is to find an optimal dispatch p_G^* that minimises a specific performance metric $J(x, p_G^*)$,where x denotes the state of the power system. A feasible dispatch $p_G \in \mathcal{U}$ ensures the power system's secure operation while aligning generation with demand.

Efforts to develop efficient algorithms for OPF have led to the introduction of several approximate methods. Most notably is the DC-OPF formulation which linearizes power flow constraints by neglecting line resistances under the assumption that line resistances are significantly lower than line reactances, combined with the approximation of a flat voltage profile and small voltage angle deviations between nodes. While the objective function is often a smooth convex function, which enables the DC-OPF to be effectively solved with standard solvers, this computational efficiency trades off against accuracy. Specifically, DC-OPF tends to under perform in heavily loaded systems and often doesn't produce feasible solutions for the original OPF-problem (Baker, 2021).To counteract these infeasibilities, DC-OPF is generally recomputed iteratively, incorporating constraint adjustments (Low, 2014).

The task faced by grid operators, ensuring that energy supply meets demand, has grown considerably more difficult due to the rapid increase of intermittent energy sources such as wind and photovoltaic systems. Unlike conventional power sources, intermittent energy systems exhibit a stochastic energy production, with an output dependent on weather conditions as well as the time of the day. Consequently, to adapt to the increased randomness of injected power, stochastic optimization models have been proposed in literature. Nonetheless, the underlying difficulties in solving OPF remains in stochastic formulations. This is why the proposed methods found in literature employ the DC-OPF approximation or convex relaxations to reduce the computational burden,

This increase in complexity calls for more computational efficient algorithms, capable of solving the OPF in short timescales (Tang et al., 2017).. Rapid results from OPF is not only essential for real-time grid management but also for addressing economic dispatch problems in electricity markets with short market time units.

In recent years, the prowess of machine learning,

[a] https://orcid.org/0000-0002-2028-9847
[b] https://orcid.org/0000-0003-3111-4820

Marcial, A. and Perninge, M.
An Unsupervised Neural Network Approach for Solving the Optimal Power Flow Problem.
DOI: 10.5220/0012187400003543
In *Proceedings of the 20th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2023) - Volume 1*, pages 214-220
ISBN: 978-989-758-670-5; ISSN: 2184-2809

and in particular deep neural networks (DNNs) have found a wide range of applications from the typical classification problem to applications in optimization theory, aimed at curtailing the on-line computational burden. Within the research aimed towards optimal power flow, a number of recent papers have utilized DNNs in an effort to solve the OPF problem.

(Zhang and Zhang, 2022) does not directly provide predictions for the OPF, instead, they employ a supervised learning approach to forecast an appropriate warm-start for a conventional non-linear solver. Similarly, (Fioretto et al., 2020) adopts a supervised methodology, utilizing a lagrangian formulation to enforce the constraints. Meanwhile, in (Huang et al., 2021) while a supervised approach is presented no effort is done during training to enforce the constraints, instead they develop a post-processing step to enhance the feasibility of the neural network's predictions.

This paper's primary contribution is the introduction of an unsupervised methodology for predicting the solutions to the OPF problem. Moreover, we detail the utilization of a custom loss function, leveraging the augmented lagrangian method to train the neural network towards optimal solutions. Consequently, our approach removes the dependence of extensive datasets of labeled data.

2 OPTIMAL POWER FLOW

In this section we describe the underlying model used in Optimal Power Flow as well as presenting the general AC-OPF formulation.

2.1 Power System Model

Power systems are modeled at the system level as buses interconnected by transmission lines, with generators and loads connected to a subset of these buses. A power system can thus be conveniently modelled as a graph $\mathcal{P} = (\mathcal{N}, \mathcal{B})$, where $\mathcal{N} \subset \mathbb{N}$ represents the set of edges connecting the nodes. We will adhere to the naming conventions of power systems, referring to $\mathcal{N}$ as the set of nodes and $\mathcal{B}$ the set of transmission lines. In modelling the power system, two equivalent formulations can be used (Low, 2014): one where $\mathcal{P}$ is a directed graph and the other where $\mathcal{P}$ is an undirected graph. In this paper, we have chosen the latter formulation; thus if $(k,l) \in \mathcal{B}$ then $(l,k) \in \mathcal{B}$. For an n-bus system, we assume, without loss of generality, that $\mathcal{N} = \{0,...,n-1\}$. Additionally, there is a bus $r_B \in \mathcal{N}$ which we refer to as the reference bus.

Each transmission line connecting two buses has the admittance $y_{i,k} = g_{i,k} + jb_{i,k},\ (i,k) \in \mathcal{B}$. There may also be an impedance connecting a bus to ground, denoted by $y_{s,i}$ for $i \in \mathcal{N}$

The power system has a set of generators, indexed by $\mathcal{G} \subseteq \mathcal{N}$. For each $i \in \mathcal{G}$, the associated generator can produce active power $p_{gi} \in \mathbb{R}_+$ and reactive power $q_{gi} \in \mathbb{R}$. Let $p_G := (p_{gi})_{i\in\mathcal{G}} \in \mathbb{R}_+^{|\mathcal{N}|}$ represent the vector of power generated in the power system. Similarly, let $q_G := (q_{gi})_{i\in\mathcal{G}} \in \mathbb{R}^{|\mathcal{N}|}$ be the vector of reactive power produced by the generators in the system. Sometimes, it's more practical to work with the apparent power produced by the generators. In that case, let $s_G := (s_{gi})_{i\in\mathcal{G}} \in \mathbb{C}^{|\mathcal{N}|}$, where $s_{gi} = p_{gi} + jq_{gi}$.

Unlike the generators, each bus can be characterized by the load $s_{di} = p_{di} + jq_{di}$, noting that for some $i \in \mathcal{N}$, $s_{di} = 0 + j0$. To identify the non-zero loads, we define $\mathcal{D}_p := \{i \in \mathcal{N} : p_{di} \neq 0\}$ and $\mathcal{D}_q := \{i \in \mathcal{N} : q_{di} \neq 0\}$. Further, let $p_D := (p_{di})_{i\in\mathcal{D}_p}$ and $q_D := (q_{di})_{i\in\mathcal{D}_q}$.

In terms of bus-related variables, let $V := (v_i)_{i\in\mathcal{N}} \in \mathbb{C}^{|\mathcal{N}|}$ denote the voltage at each bus, where $v_i = V_i e^{j\delta_i}$. We further define the vectors of voltage magnitude and angles as $|V| := (V_1,...,V_n) \in \mathbb{R}^{|\mathcal{N}|}$ and $\angle V := (\delta_1,...,\delta_n) \in [-\pi,\pi]^{|\mathcal{N}|}$ respectively. We assume that $\delta_{r_B} = 0$ and that $\delta_i,\ i \in \mathcal{N}\backslash\{r_B\}$ is measured with r_B as reference.

Assuming that $|\mathcal{N}| = n$, we define $S := (s_1,...,s_n) \in \mathbb{C}^{|\mathcal{N}|}$ as the vector of net apparent power injection in each node of the power system. The nodal apparent power injection is determined using Kirchoff's law:

$$s_i = v_i \sum_{k:(i,k)\in\mathcal{B}} y_{i,k}^* (v_i^* - v_k^*) + |v_i|^2 y_{si}, \quad i \in \mathcal{N} \tag{1}$$

The net apparent power injection in a node is equivalent to the difference between the scheduled generation s_{gi} and the scheduled load s_{di};

$$s_i = \begin{cases} s_{gi} - s_{di} & i \in \mathcal{G} \\ -s_{di} & i \in \mathcal{N}\backslash\mathcal{G}. \end{cases} \tag{2}$$

Note that, if $i \notin \mathcal{D}_p \cup \mathcal{D}_q$ then $s_{di} = 0 + j0$

Generators have both lower and upper limits in terms of active and reactive power output. For a generator located at bus $i \in \mathcal{G}$, the range of its active power output is constrained to $p_{gi} \in [P_{gi}^{min}, P_{gi}^{max}]$. Similarly the reactive power is constrained to the interval $q_{gi} \in [Q_{gi}^{min}, Q_{gi}^{max}]$.

To ensure stable operation of the power system, it's imperative that the voltages across the buses remain within acceptable levels. Consequently, the voltage magnitude for bus $i \in \mathcal{N}$ is confined to the interval $[V_i^{min}, V_i^{max}]$.

We define the restrictions on the generators capabilities and bus voltage limits to be the sets

$$\mathcal{A} := \prod_{i\in\mathcal{G}} [P_{gi}^{min}, P_{gi}^{max}] \tag{3}$$

$$\mathcal{R} := \prod_{i\in\mathcal{G}} [Q_{gi}^{min}, P_{gi}^{max}] \tag{4}$$

$$\mathcal{V} := \prod_{i\in\mathcal{N}} [V_{i}^{min}, V_{i}^{max}] \tag{5}$$

2.2 Optimization Model

In this paper, the OPF formulation we employ focuses on economic dispatch. Specifically the objective is to identify the operating point for generators that results in the most cost efficient production. Given a cost function $C : \mathcal{A} \to \mathbb{R}$, the OPF can be expressed as

$$\min_{V} \; C(p_G) \tag{6}$$

$$\text{s.t.} \quad \mathbf{g}(V,S) = \mathbf{0} \tag{7}$$

$$p_G \in \mathcal{A} \tag{8}$$

$$q_G \in \mathcal{R} \tag{9}$$

$$|V| \in \mathcal{V}. \tag{10}$$

While there exist various ways to formulate the cost function given by equation (6), polynomial and piecewise linear approximations are frequently encountered in literature. For the purpose of this paper, we opt for the polynomial approach. Consequently, the cost function is expressed as:

$$C(p_G) := \sum_{i\in\mathcal{G}} (C_2^i p_{gi}^2 + C_1^i p_{gi} + C_0^i) \tag{11}$$

Note that, the specific formulation of the cost function does not limit the applicability of this paper; the proposed methodology can accommodate a diverse range of formulations.

The power balance at each bus within the power system is represented by equation (7) and is defined as:

$$g_i(V,S) = f_i(V) - s_i, \quad i \in \mathcal{N} \tag{12}$$

where $f_i(V)$ corresponds to the right-hand side of (1)

3 DNN APPROXIMATION OF OPTIMAL SOLUTION

3.1 Function Approximation

In this paper, we assume that the power system described by $\mathcal{P}$ is static, meaning the network topology remains unaltered. Therefore, given $(p_D, q_D) \in \mathcal{D}_p \times \mathcal{D}_q$, the objective is to determine the optimal voltage vector V^* (if it exists) that solves the OPF instance defined by (p_D, q_D). With this context, the OPF can be perceived as an operator that maps a given instance defined by the loading into an optimal solution, (Zhou et al., 2020),(Falconer and Mones, 2022)

$$F : \mathcal{P}_D \times Q_D \to \mathbb{C}^{|\mathcal{N}|} \tag{13}$$

here, $\mathcal{P}_D$ and Q_D represent sets of active and reactive loads, respectively.

For the development of the proposed approximation of the solution to the OPF, it's advantageous to modify the standard OPF formulation. Consider the power flow equations in (1), they can be divided into two sets of equations,

$$s_{gi} = s_i + s_{di}, \quad i \in \mathcal{G} \tag{14}$$

$$0 = s_i + s_{di}, \quad i \in \mathcal{N}\setminus\mathcal{G}. \tag{15}$$

Combining (14) with (8)-(9) yields the inequalities

$$P_{gi}^{min} \le Re(s_i + s_{di}) \le P_{gi}^{max} \tag{16}$$

$$Q_{gi}^{min} \le Im(s_i + s_{di}) \le Q_{gi}^{max} \tag{17}$$

Further we define the set

$$\mathcal{U}(p_D, q_D) := \{V \in \mathbb{C}^{|\mathcal{N}|} : (16),(17),(10) \text{ feasible}\}. \tag{18}$$

Then, given $s_D \in \mathcal{P}_D \times Q_D$ and $V \in \mathcal{U}(s_D)$ an operating point (p_G, q_G) can be recovered from (7).

Evaluating F using conventional iterative solvers is cumbersome. To reduce the computational burden at inference time we therefore propose the use of a neural network, denoted $\hat{F}(p_D, q_D; \theta)$ to approximate (13), where θ is the parameters that are learned during training of the neural network.

A major problem is to ensure that $\hat{F}(p_D, q_D; \theta) \in \mathcal{U}(p_D, q_D)$. Lagrangian relaxation is a common method in conventional constrained optimization and it has been used in training of neural networks for constrained optimization problems. In (Zhang and Zhang, 2022) it is used to solve supervised AC-OPF using a two networks for predicting the primal and dual variables respectively. In (Pan et al., 2020), (Pan et al., 2022) a penalty function is used as a regularization term in the loss function for penalizing deviations from the constraints.

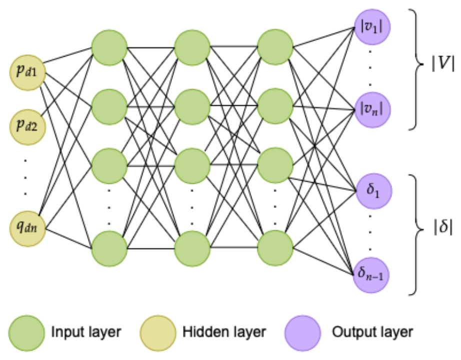

Figure 1: Neural network architecture.

3.2 Unsupervised Learning

To train $\hat{F}(\cdot;\theta)$ using supervised learning, one requires a substantial dataset of load samples, denoted as $\mathcal{D}_x = \{(p_D^i, q_D^i)\}_{i=1}^{\Gamma}$, where $(p_D^i, q_D^i) \in \mathcal{P}_D \times \mathcal{Q}_D$. Correspondingly, there is a set $\mathcal{D}_y = \{y^i\}_{i=1}^{\Gamma}$ representing the ground truth, with $y^i = F(p_D^i, q_D^i)$. The neural network is then trained using some loss function, e.g. the empiric risk, as shown in (19). Regularization terms can further be incorporated to penalize any constraint violations. Equation (19) will be revisited in subsequent sections, particularly when comparing the supervised and unsupervised learning methodologies.

$$\min_{\theta} \frac{1}{\Gamma}\sum_{i=1}^{\Gamma} \|\hat{F}(x^i;\theta) - y^i\|_2^2, \tag{19}$$

Given that $y_i \in \mathcal{U}(s_D^i)$, the network is trained to imitate the true solution. Under an unsupervised framework, the label set $\mathcal{D}_y$ is eliminated. This means the network need to find an optimal θ^* such that $\hat{F}(p_D, q_D; \theta^*) \approx F(p_D, q_D)$, without the guidance of ground truth. In essence, for each sample in $\mathcal{D}_x$ the neural network need to predict a feasible voltage that minimize the cost. The unsupervised training can therefore be formulated as the constrained optimization problem

$$\begin{aligned} \arg\min_{\theta} \quad & \sum_{i=1}^{\Gamma} C(\hat{F}(S_D^i;\theta)) \\ \text{s.t.} \quad & \hat{F}(p_D^i, q_D^i;\theta) \in \mathcal{V}(S_D^i) \quad \forall (p_D^i, q_D^i) \in \mathcal{D}_x \end{aligned} \tag{20}$$

Similar to approaches used in conventional optimization algorithms, (20) is in this paper relaxed to an unconstrained optimization problem. Specifically, we will use the Augmented Lagrangian Method (ALM) (Bertsekas, 2016) which is a method that combines the Lagrangian method and penalty methods. Previous works in approximating constrained optimization problems using machine learning have reported ALM successful. In particular, ALM has been used in physics informed neural networks such as in (Djeumou et al., 2022) (Lu et al., 2021) (Dener et al., 2020)

3.3 Neural Network Architecture

Consistent with the previous cited works of using machine learning for OPF, the neural network architecture employed in this paper is a feed-forward architecture, (Figure 1), as it has shown good performance in supervised settings. For the unsupervised approach, a single neural network is utilized. However, for the supervised approach, two neural networks are deployed: one to predict voltage magnitudes and another for voltage angles.

As highlighted in section 3.1, the neural network's input comprises the load $(p_D, q_D) \in \mathbb{R}^{|\mathcal{N}|} \times \mathbb{R}^{|\mathcal{N}|}$. This configuration corresponds to an input layer in the neural network with $2|\mathcal{N}|$ neurons. However, it is worth noting that since generally $\mathcal{N} \setminus (\mathcal{D}_p \cup \mathcal{D}_q) \neq \emptyset$, we can effectively reduce the neuron count in the input layer to $|\mathcal{D}_p| + |\mathcal{D}_q|$.

Subsequent to the input layer, there is a sequence of hidden layers, L_i, with each layer comprising n_i neurons. Both the depth (number of layer) and width (neurons per layer) of this structure are considered hyperparameters, that are problem specific. An increasing size of $\mathcal{N}$ benefits from an increasing number of neurons per layer. In the specific architecture used in this paper, all activation function within the hidden layers are Rectifier Linear Units (ReLU).

The output layer is designed with $2|\mathcal{N}| - 1$ neurons. Out of these, $|\mathcal{N}|$ neurons are allocated for representing the voltage magnitude V, and the remainder serves as outputs for the voltage angle, recalling that the bus angle of bus n_{r_B} is assumed known.

One strategy to integrate a subset of the constraints into the architecture of the neural network is to use appropriate activation functions on the output layer of the neural network. Specifically, let $\mathbf{x} = (\mathbf{x}_v, \mathbf{x}_\delta)$ be the output vector of the neural network, corresponding to the voltage magnitudes and angles respectively. Then using e.g. the sigmoid activation function we obtain the mapping $\sigma(\mathbf{x}) = (\sigma_v(\mathbf{x}_v), \sigma_\delta(\mathbf{x}_\delta)) \in [0,1]^{|\mathcal{N}|} \times [0,1]^{|\mathcal{N}|-1}$. The actual voltage magnitude and angles can then be obtained by

$$\sigma_V(x_v) \mapsto (V_i^{min})_{i\in\mathcal{N}} + \left(\sigma_{v,i}(x_i)(V_i^{max} - V_i^{min})\right)_{i\in\mathcal{N}} \tag{21}$$

and the corresponding angles are obtained as

$$\sigma_\delta(x_\delta) \mapsto (\delta_i^{min})_{i\in\mathcal{N}\setminus\{n_R\}} + \left(\sigma_i(x_i)(\delta_i^{max} - \delta_i^{min})\right)_{i\in\mathcal{N}\setminus\{n_{r_B}\}} \tag{22}$$

Through this methodology, the neural network effectively enforces the voltage constraints.

3.4 Loss Function

To solve (20), ALM is used to relax the constrained optimization problem into an unconstrained problem. In general, ALM algorithm consists of solving a sequence of sub-optimization problems

$$x_k^* = \arg\min_x \mathcal{L}^k(x, \lambda_k, \mu_k, \rho_k) \tag{23}$$

with,

$$\begin{aligned}\mathcal{L}^k(\mathbf{x}, \lambda_k, \mu_k, \rho_k) &= f(x) + \lambda_k^T \mathbf{g}(x) + \mu_k^T \mathbf{h}(x) + \\ &+ \frac{\rho_k}{2}\|\mathbf{h}(\mathbf{x})\|_2^2 + \frac{\rho_k}{2}\|\phi(\mathbf{h}(\mathbf{x}))\|_2^2\end{aligned} \tag{24}$$

Where, λ_k and μ_k are Lagrange multipliers and the norms serves as penalty terms, weighted by the penalty parameters ρ_k, and $\phi(\mathbf{h}(\mathbf{x}) = \max\{0, \mathbf{h}(\mathbf{x})\}$. Then, under some mild regularity conditions it can be shown that $\lim_{k\to\infty}\{x_k^*\} = x^*$, solves the original optimization problem.

In practice, each sub-problem is finitely iterated until a stopping criteria $||\nabla_x \mathcal{L}|| < \varepsilon_k$ (Bertsekas, 2016), however in the training of the neural network we have chosen a slight different approach were we instead use stochastic gradient descent until each sub-optimization problem stops improving, which is covered in the next section.

3.5 Training

The load samples set, $\mathcal{D}_x$ is divided into two disjoint sets $\mathcal{D}_\mathcal{T}$ and $\mathcal{D}_\mathcal{E}$ out of which the former is used for training and the latter for evaluating the performance of the trained network.

In the spirit of ALM, the training of $\hat{F}$ involves solving a sequence of optimization problems. For each outer iteration $k \in \{0, ..., K-1\}$, the goal is to determine the optimal parameter θ_k corresponding to sub-optimization k. Within each outer-iteration, the neural network is trained using the Adam optimizer (Kingma and Ba, 2017). The training process follows the commonly used mini-batch approach, where the training set is randomly segmented into mini-batches $\mathcal{B}_i \subset \mathcal{D}_\mathcal{T}$. Each mini-batch is used to compute the gradient and subsequent weight updates for the network. The entire dataset is processed in this fashion up to n_{epoch} times.

In an effort to avoid over fitting and reduce training time, an early stopping mechanism with a preset patience is employed. The loss function $\mathcal{L}_k$ is evaluated after each epoch. If no improvement is recorded for a fixed number of consecutive epochs, the training of the k-th optimization problem is halted. Additionally, adaptive learning rates are used, reducing the learning rate if there is no observed improvement in loss. The learning rate is reverted back to the initial learning rate in each new outer iteration. Once an inner iteration is completed, the Lagrange multipliers are updated in accordance with the ALM algorithm:

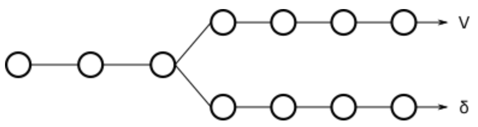

Figure 2: Neural network configuration for unsupervised learning.

$$\begin{aligned}\lambda_{k+1} &= \lambda_k + \rho_k g(x_k^*) \\ \mu_{k+1} &= \max\{0, \rho_k h(x_k^*)\}\end{aligned} \tag{25}$$

```
Data: Training dataset $\mathcal{D}$
Result: optimal parameter $\theta^*$
$\lambda_0 \leftarrow 0$;
$\mu_0 \leftarrow 0$;
$\rho_0 \leftarrow \rho_{init}$;
for $k \leftarrow 0$ to $K-1$ do
    minimize $\mathcal{L}^k(\theta, \lambda_k, \mu_k, \rho_k)$ using Adam.;
    $\lambda_{k+1} \leftarrow \lambda_k + \rho_k g(\hat{F}(S_D; \theta_k)$;
    $\mu_{k+1} \leftarrow \max\{0, \mu_k + \rho_k h(\hat{F}(S_D; \theta_k)\}$;
    if $\|h(x)\|_\infty > \eta\|h^*\|$ then
        $h^* \leftarrow \|h(x)\|_\infty$
        $\rho_{k+1} \leftarrow \min\{\gamma\rho_k, \rho_{max}\}$
    end
end
```

Algorithm 1: Training the neural network.

4 NUMERICAL RESULTS

4.1 Hyperparameters

The proposed approach is conducted on the IEEE 300-bus system. The neural network is developed using the pyTorch library and trained on a MacBook pro 2 GHz Quad-Core Intel Core i5, 16 GB RAM.

For the supervised approach, both neural networks feature an architecture with four hidden layers, with a sequentially number of neurons: 1024,768,512 and 256. Meanwhile, for the unsupervised approach, the hyperparameters with respect to hidden layers has a network configuration as illustrated in Figure 2. This configuration comprises hidden layers with neuron counts of 2048,1024,768,512 and 256.

For the training dataset, we utilize a total of $|\mathcal{D}| = 30000$ samples of loads. The loads exhibit correlations. The outer-loop is executed for 50 iterations, while the inner iteration involves up to 50 epochs. To facilitate learning, we employ an exponential decay strategy for the learning rate with a decay factor of $\gamma = 0.9$. The learning rate is decayed if the loss does not improve for 5 consecutive epochs. We initialize the learning rate at 10^{-4}. Furthermore, we set the early stopping patience to 10 epochs, allowing for timely termination if necessary. Regarding the ALM algorithm, we set the initial penalty parameter, ρ_0 to 500, while the Lagrange multipliers are initialized as zeros.

4.2 Performance

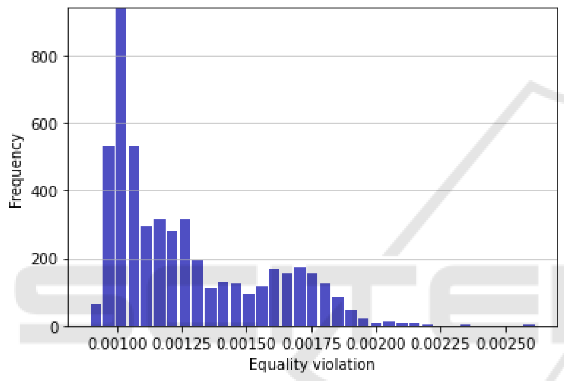

Figure 3: Mean violation of the equality constraints.

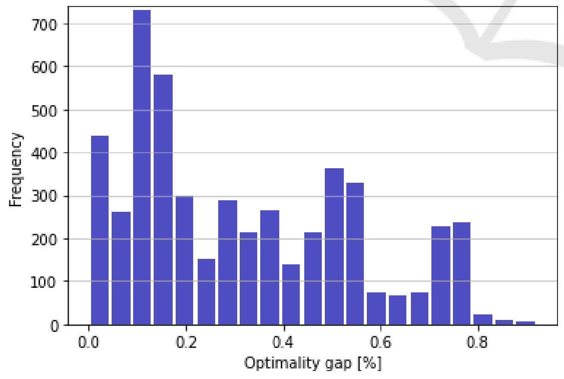

Figure 4: Optimality gap of the objective function.

The dataset $\mathcal{D}$ was divided into a training set $\mathcal{D}_T$ and a test set $\mathcal{D}_E$ with the proportion $|\mathcal{D}_T = 30000$ and $|\mathcal{D}_E| = 5000$. The dataset used is labeled with the optimal solution retrieved from Matpower (Zimmerman et al., 2011)(Wang et al., 2007). The unsupervised approach was trained using the Mean Squared Error as loss function, without any regularization terms added. The results obtained using the test set are comprehensively displayed in Table 1. Histograms illustrating the equality violation as and the optimality gap, when using the unsupervised approach, can be seen in Figure 3-4 respectively.
Using the labeled data, the optimality gap is computed as

$$O_{gap} = 100\frac{\hat{F}(s_D;\theta^*) - F(s_D)}{F(s_D)}. \qquad (26)$$

The values reported in Table 1 are the average across the entire test set, except the maximum equality constraint violation which shows the maximum across the entire test set.

Table 1: Result of supervised and unsupervised training.

	Unsupervised	Supervised
Opt.gap [%]	0.316	0.216
Eq.	$1.3\cdot10^{-3}$	$2.6\cdot10^{-3}$
Ineq.	$6.7\cdot10^{-7}$	$5.4\cdot10^{-6}$
Max eq.	0.023	0.068

5 CONCLUSION

The need for fast and accurate solutions of the Optimal Power Flow problem is becoming more important as the power systems are experiencing more unpredictability in the form of intermittent power. While training neural networks do not offer a lighter computational workload compared to conventional iterative non-linear solvers, they have a distinct advantage in that they shift the computational load off-line. A trained neural network outputs the solution in negligible time. This makes neural network a promising candidate for time sensitive applications.

Although the computational burden of supervised learning is offline, it comes with its own challenges. It demands a considerable amount of labeled data. This translates to the need for a repository of OPF solutions coming from solver like the interior-point method.

This paper primarily delved into the application of unsupervised learning for addressing the AC-OPF problem, without considering post-processing. A logical extension to this paper would be to explore other neural network architectures. The feed-forward architecture, as employed in this paper, might encounter challenges when confronted with significantly larger power grid models. Hence it is therefore interesting to investigate other architectures that scales well with increasingly complex models.

REFERENCES

Baker, K. (2021). Solutions of dc opf are never ac feasible. In *Proceedings of the Twelfth ACM International Conference on Future Energy Systems*, pages 264–268.

Bertsekas, D. (2016). *Nonlinear Programming*, volume 4. Athena Scientific.

Capitanescu, F. (2011). Assessing reactive power reserves with respect to operating constraints and voltage stability. *IEEE Transactions on Power Systems*, 26(4):2224–2234.

Carpentier, J. (1962). Contribution to the economic dispatch problem. *Bulletin de la Societe Francoise des Electriciens*, 3(8):431–447.

Dener, A., Miller, M. A., Churchill, R. M., Munson, T., and Chang, C.-S. (2020). Training neural networks under physical constraints using a stochastic augmented lagrangian approach. *arXiv preprint arXiv:2009.07330*.

Djeumou, F., Neary, C., Goubault, E., Putot, S., and Topcu, U. (2022). Neural networks with physics-informed architectures and constraints for dynamical systems modeling. In *Learning for Dynamics and Control Conference*, pages 263–277. PMLR.

Falconer, T. and Mones, L. (2022). Leveraging power grid topology in machine learning assisted optimal power flow.

Fioretto, F., Mak, T. W., and Van Hentenryck, P. (2020). Predicting ac optimal power flows: Combining deep learning and lagrangian dual methods. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 630–637.

Huang, W., Pan, X., Chen, M., and Low, S. H. (2021). Deepopf-v: Solving ac-opf problems efficiently. *IEEE Transactions on Power Systems*, 37(1):800–803.

Kingma, D. P. and Ba, J. (2017). Adam: A method for stochastic optimization.

Lavaei, J. and Low, S. H. (2011). Zero duality gap in optimal power flow problem. *IEEE Transactions on Power Systems*, 27(1):92–107.

Low, S. H. (2014). Convex relaxation of optimal power flow—part i: Formulations and equivalence. *IEEE Transactions on Control of Network Systems*, 1(1):15–27.

Lu, L., Pestourie, R., Yao, W., Wang, Z., Verdugo, F., and Johnson, S. G. (2021). Physics-informed neural networks with hard constraints for inverse design. *SIAM Journal on Scientific Computing*, 43(6):B1105–B1132.

Pan, X., Chen, M., Zhao, T., and Low, S. H. (2022). Deepopf: A feasibility-optimized deep neural network approach for ac optimal power flow problems.

Pan, X., Zhao, T., and Chen, M. (2020). Deepopf: Deep neural network for dc optimal power flow.

Skolfield, J. K. and Escobedo, A. R. (2022). Operations research in optimal power flow: A guide to recent and emerging methodologies and applications. *European Journal of Operational Research*, 300(2):387–404.

Tang, Y., Dvijotham, K., and Low, S. (2017). Real-time optimal power flow. *IEEE Transactions on Smart Grid*, 8(6):2963–2973.

Wang, H., Murillo-Sanchez, C. E., Zimmerman, R. D., and Thomas, R. J. (2007). On computational issues of market-based optimal power flow. *IEEE Transactions on Power Systems*, 22(3):1185–1193.

Zhang, L. and Zhang, B. (2022). Learning to solve the ac optimal power flow via a lagrangian approach. In *2022 North American Power Symposium (NAPS)*, pages 1–6. IEEE.

Zhou, F., Anderson, J., and Low, S. H. (2020). The optimal power flow operator: Theory and computation.

Zimmerman, R. D., Murillo-Sánchez, C. E., and Thomas, R. J. (2011). Matpower: Steady-state operations, planning, and analysis tools for power systems research and education. *IEEE Transactions on Power Systems*, 26(1):12–19.