Solver-Grade AC-OPF at Near-DC Speed: Residual Graph Neural Networks with Newton-Raphson Restoration

Jarah Alqasim, Ahmad Bin Afzal, Charalambos Konstantinou

jarah.alqasim@kaust.edu.sa, ahmadbin.afzal@kaust.edu.sa, charalambos.konstantinou@kaust.edu.sa

Abstract

Optimal power flow (OPF) sets the operating point of an electrical grid. Its exact alternating-current (AC) form is nonconvex and too slow for near-real-time use. The linear direct-current (DC) approximation is fast, but it returns operating points that are not physically valid on the AC grid. We reproduce and extend a residual-learning framework that maps a DC-OPF solution to its AC counterpart with a topology-aware graph neural network (GNN), training on 270k solved instances per system from the OPFData benchmark. We then add a Newton-Raphson feasibility-restoration layer that makes the prediction satisfy the AC power-flow equations exactly. On the IEEE 118-bus system, the residual GNN reproduces the original study's reported bus-voltage error to within 12%, and improves on our graph-convolutional and graph-attention baselines by about an order of magnitude. On two systems the original work never tested, IEEE 14 and IEEE 30, residual learning improves the voltage error by factors of 15 to 34 across the three architectures. The restoration layer then drives the average AC power-flow residual from about 10^{-2} down to about 10^{-6} p.u., which is indistinguishable from a classical solver, at roughly 3.3× the speed of an interior-point AC-OPF solve. Because we clip the predictions to generator limits, the same layer also improves the economics of the solution, lowering the dispatch error on every system and cutting the cost gap to the AC-OPF optimum from 0.60% to 0.06% on IEEE 118. Physics-restored residual learning therefore bridges DC and AC optimal power flow. It delivers solver-grade, AC-feasible operating points at near-DC speed.

Date: August 19, 2026

Code: https://github.com/jarahalqasim-spec/Feasible-AC-OPF-via-Residual-GNNs-and-NRR

Contents

1 Introduction 2

2 Related Work 2

3 Methodology 3

3.1 Overview . . . 3

3.2 AC and DC optimal power flow . . . 4

3.3 Residual learning . . . 4

3.4 Topology-aware graph neural network . . . 5

3.5 Newton-Raphson feasibility restoration . . . 5

3.6 Evaluation procedure . . . 7

4 Results 8

4.1 Residual learning across systems . . . 8

4.2 Feasibility restoration . . . 9

4.3 Runtime . . . 10

5 Discussion 12

6 Broader Impact: Saudi Arabia and Vision 2030 13

7 Limitations and Future Work 14

8 Conclusion 15

References 15

1 Introduction

Every few minutes, a power-system operator must decide how much electricity each generator should produce. The goal is to meet demand at the lowest cost without violating any physical limit. This decision is the *optimal power flow* (OPF) problem, and it underpins economic dispatch, market clearing, and security assessment across the grid [1, 2]. The most accurate version is alternating-current OPF (AC-OPF). It models the full nonlinear physics of the grid: bus voltage magnitudes, phase angles, reactive power, and line losses. This accuracy is expensive. AC-OPF is nonconvex and NP-hard, so the interior-point solvers used for it become slow and sometimes intractable on large grids that must run in real time [2].

To stay fast, operators often use the direct-current approximation (DC-OPF). DC-OPF linearizes the physics: it assumes every voltage magnitude equals one and ignores reactive power and losses [1]. It solves in milliseconds. Its solutions are not physically valid for the AC grid, however, because substituting a DC solution back into the AC equations leaves large power imbalances. This trade-off between speed and physical accuracy is becoming more pressing as grids add variable renewable generation and distributed energy resources (DERs). Such resources change quickly and make voltage and reactive-power management harder [3–5]. Modern operations need OPF solutions that are fast *and* AC-feasible at the same time.

Machine learning offers one route to this goal by moving the heavy computation offline. A model trained on many solved instances can predict an operating point almost instantly [2]. A recent framework by Za'ter et al. [6] does not learn the full AC solution from scratch. Instead, it learns only the *residual*: the difference between a fast DC-OPF solution and the true AC solution. The model is a graph neural network (GNN) whose structure mirrors the grid, evaluated on the IEEE 57- and 118-bus systems and a 2000-bus case. Anchoring the prediction to a physics-based DC baseline makes training more stable and helps the model generalize. The framework still has one limitation. Its raw outputs satisfy the AC power-flow equations only approximately, so the predicted operating point is not guaranteed to be feasible.

Throughout, we assume a single-period, static OPF on a fixed grid topology, and we train in a supervised setting on a large public dataset of solved AC-OPF instances. Within that scope, this report reproduces the residual-learning framework and extends it. Our contributions are threefold. First, we independently reproduce the method on the IEEE 118-bus system at full data scale, and we measure how much residual learning improves accuracy over standard graph-network baselines. Second, we extend the study to the IEEE 14- and 30-bus systems, which the original work did not test, and we show how the benefit of residual learning changes with system size. Third, and most importantly, we add a *Newton-Raphson feasibility-restoration layer*. It takes the network's near-optimal prediction as a starting guess and, in a few iterations, makes it satisfy the AC power-flow equations exactly. The result is a pipeline that returns solver-grade, AC-feasible operating points at near-DC speed.

2 Related Work

Classical OPF formulations and solvers. Optimal power flow has a long history, and several surveys catalogue its formulations and solution methods [1, 2, 7]. Frank and Rebennack [1] give a tutorial treatment of the AC and DC formulations and of the interior-point methods used to solve them. Babiker et al. [2] review the state of the art, from convex relaxations to metaheuristics and, increasingly, learning-based surrogates; they identify the trade-off between speed and feasibility as a central open challenge. For distribution systems, Yang et al. [7] survey the main formulations and highlight the role of linearized power flow in keeping the problem tractable.

Approximations and heuristic solvers. A large body of work trades physical accuracy for speed. Ahmadi and Martí [8] pose distribution-system optimization on a linearized power-flow model, one instance of the linear approximation that DC-OPF represents, while others use metaheuristics such as sunflower optimization [9] or optimize the renewable mix directly [5]. These methods give useful operating points, but none guarantees AC feasibility at real-time speed, which motivates learning the correction from a trusted baseline instead of re-solving the problem from scratch.

Motivation from renewable and distributed resources. Caballero-Peña et al. [3] review DER modelling and its impact on distribution networks, and Lee [4] reviews the operational and economic challenges DER integration poses for a system operator. Both stress that reactive power, voltage stability, and congestion become increasingly binding as inverter-based resources spread. These are exactly the quantities that DC-OPF ignores.

Graph neural networks for OPF. Because a power grid is naturally a graph, GNNs have become a standard surrogate architecture. Owerko et al. [10] first applied graph convolutional networks to OPF, predicting set-points directly from the network topology. Zhang et al. [11] extended this with graph attention, weighting each neighbour by its electrical influence rather than treating all connections alike. Khaloie et al. [12] survey this literature and define the feasibility-distance metric we adopt. We use the first two architectures as our baselines, following the original study.

Feasibility restoration and warm starts. A parallel line of work accepts that learned predictions are imperfect and repairs them. Donti et al. [13] complete partial predictions and correct them with gradient steps to satisfy hard constraints, and Baker [14] and Diehl [15] instead use predictions to warm-start an AC-OPF solve, cutting solver iterations. Our restoration layer follows the same logic but changes the target: rather than warm-starting an *optimization*, we warm-start a *power flow*, which has no objective to minimize and is therefore far cheaper, while still enforcing the AC equations exactly.

Learning-based OPF and the residual paradigm. Within this line of work, Za'ter et al. [6] propose the residual-learning framework that we reproduce and extend. A topology-aware GNN learns the correction from a DC-OPF solution to its AC counterpart, trained with a physics-aware loss. Framing the task as residual correction, rather than end-to-end prediction, lowers the data requirement and improves generalization. The main gap in that work, and in learning-based OPF more broadly, is that raw network outputs are only approximately feasible. Our contribution closes this gap with an explicit Newton-Raphson restoration step, and it adds evidence on the smaller systems that the original study did not examine.

3 Methodology

3.1 Overview

Our pipeline turns a fast DC-OPF solution into a feasible AC-OPF operating point in three stages, shown in Figure 1. First, a conventional solver computes the DC-OPF solution. Second, a topology-aware graph neural network predicts the residual correction that carries the DC solution to its AC counterpart. Third, a Newton-Raphson power flow restores exact feasibility, using the network's prediction as a starting guess. The first two stages reproduce the framework of Za'ter et al. [6]. The third stage is our addition, and it is what turns an approximate prediction into an operating point that satisfies the AC equations exactly.

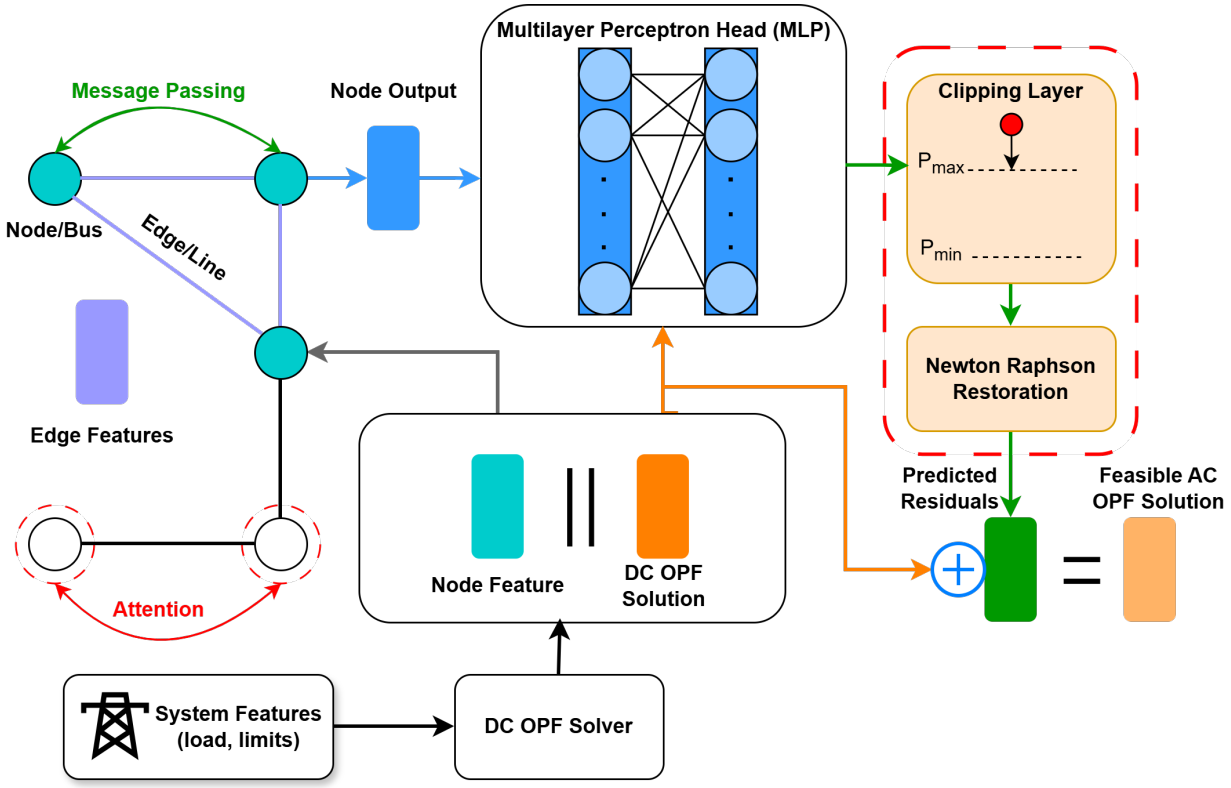


Figure 1 The proposed pipeline.

3.2 AC and DC optimal power flow

Let $\mathcal{B}$ be the set of buses and $\mathcal{L} \subseteq \mathcal{B} \times \mathcal{B}$ the set of branches. Every bus b carries four quantities, which we use throughout the report:

- u_b, θ_b: the voltage magnitude and angle;
- $p_{g,b}$, $q_{g,b}$: the generator active and reactive power;
- $p_{d,b}$, $q_{d,b}$: the active and reactive demand.

AC-OPF minimizes the generation cost subject to three families of constraints: the nonlinear nodal power balances, the voltage and generator limits, and the thermal line limits. Writing the balances explicitly,

$$\min_{x} \sum_{b\in\mathcal{B}} f_b(p_{g,b}) \quad \text{s.t.} \quad p_{g,b} - p_{d,b} = \sum_{n:(b,n)\in\mathcal{L}} p_{bn}(u,\theta), \;\; q_{g,b} - q_{d,b} = \sum_{n:(b,n)\in\mathcal{L}} q_{bn}(u,\theta), \tag{1}$$

where $x \equiv (p_g, q_g, u, \theta)$, and p_{bn} and q_{bn} are the nonlinear branch flows. We write $x^\star_{\mathrm{AC}}(z)$ for the optimal AC solution of an instance z (its loads, topology, and costs). DC-OPF linearizes these equations by assuming unit voltages, small angle differences, and no losses. This gives the linear relation $p = B\theta$, where B is the bus susceptance matrix. DC-OPF returns $x_{\mathrm{DC}}(z) = (p_g^{\mathrm{DC}}, \theta^{\mathrm{DC}})$ in milliseconds, but this solution is infeasible for the AC problem (1).

3.3 Residual learning

We first build an approximate AC point from the DC solution, $x^{(0)}_{\mathrm{AC}} = (p_g^{\mathrm{DC}}, q_g^{(0)} = \mathbf{0}, u^{(0)} = \mathbf{1}, \theta^{\mathrm{DC}})$, and define the residual as

$$\Delta x^\star(z) = x^\star_{\mathrm{AC}}(z) - x^{(0)}_{\mathrm{AC}}(z). \tag{2}$$

A model is trained to approximate the map $z \mapsto \Delta x^\star(z)$ defined in (2). We then reconstruct the AC prediction as $\hat{x}_{\mathrm{AC}} = x^{(0)}_{\mathrm{AC}} + \hat{\Delta} x$. Learning the residual, instead of the full solution, focuses the model's capacity on the nonlinear effects that the DC approximation misses.

3.4 Topology-aware graph neural network

The network mirrors the grid: each bus is a node, and each line or transformer is a typed edge. First, multilayer perceptrons encode the node and edge features, and the DC solution is attached to the input; we call this *local* integration. The network then applies K message-passing layers. In each layer, every bus gathers information from its electrical neighbours through an attention mechanism that weights neighbours by how strongly they interact and adds simple geometric and DC-flow biases. Each layer updates a bus's embedding with a residual, layer-normalized rule that keeps training stable. Message passing is *typed*: physically different connections (AC lines, transformers, and generator or load links) use different weights. After the message-passing layers, the network combines a pooled summary of the whole graph with the full DC solution again, which we call *global* fusion, and passes the result to the output heads. These heads predict the residuals for bus voltage and angle, for generator active and reactive power, and for branch apparent-power magnitude. Skip connections add each residual to the DC baseline, so the network only ever learns deviations from it.

We train the network by minimizing a physics-aware loss. It combines a supervised term, which matches the AC labels, with soft penalties that discourage physically invalid predictions:

$$\mathcal{L} = \mathcal{L}_{\mathrm{sup}} + \lambda_{\mathrm{pf}}\,\mathcal{L}_{\mathrm{PF}} + \lambda_{\mathrm{box}}\,\mathcal{L}_{\mathrm{box}} + \lambda_{\mathrm{obj}}\,\mathcal{L}_{\mathrm{obj}} + \lambda_{\mathrm{res}}\,\mathcal{L}_{\mathrm{res}}. \tag{3}$$

In (3), $\mathcal{L}_{\mathrm{sup}}$ matches the AC labels, $\mathcal{L}_{\mathrm{PF}}$ penalizes AC power-flow residuals, $\mathcal{L}_{\mathrm{box}}$ penalizes violations of the voltage, reactive-power, and thermal limits, $\mathcal{L}_{\mathrm{obj}}$ penalizes cost deviation, and $\mathcal{L}_{\mathrm{res}}$ regularizes the size of the residual. We use the coefficients reported by the original work (λ_{pf}=λ_{box}=1, λ_{obj}=0.1, λ_{res}=0.01).

3.5 Newton-Raphson feasibility restoration

The trained network gives a prediction that is close to optimal, but that is only *approximately* consistent with the laws of physics. If we substitute it into the AC power-flow equations, small power imbalances remain at the buses. Our main methodological contribution removes these imbalances with a single warm-started Newton-Raphson AC power flow.

Two different problems. It helps to separate two tasks that are easy to confuse. Solving an *OPF* chooses the best dispatch, that is, how much each generator should produce. Solving a *power flow* answers a narrower question: given a dispatch that is already fixed, what exact physical state does the grid settle into? A power flow has no cost to minimize and no freedom of choice. It only enforces the physics, so it is far cheaper to solve than an OPF.

Our restoration step exploits this split. The network has already made the expensive *decision* well, so we keep that decision and ask a power flow to compute the exact *state* it produces.

What is held fixed and what is solved for. Every bus is assigned a standard type, and the type decides which of its four quantities (active power p, reactive power q, voltage magnitude u, and angle θ) are known and which are unknown. Table 1 records the split we use. Two quantities per bus are always fixed and two are always free, so the resulting system is square and has as many equations as unknowns.

Table 1 Hand-off from the network to the power flow. Slack and PV buses take their fixed quantities from the network prediction; PQ buses take theirs from the instance data.

Bus type	Fixed	Solved for	Count (IEEE 14)
Slack (reference)	$u, \theta = \theta^{\text{ref}}$	p, q	1
PV (generator)	p set-point, u set-point	θ, q	4
PQ (load)	$p = -p_d, q = -q_d$	θ, u	9

The equations to be satisfied. Let $Y = G + \mathrm{j}B$ be the bus admittance matrix built from the branch data, and let $\theta_{ik} = \theta_i - \theta_k$. The power that the network physically injects at bus i is

$$P_i(u, \theta) = u_i \sum_k u_k \left(G_{ik} \cos\theta_{ik} + B_{ik} \sin\theta_{ik}\right), \tag{4}$$

$$Q_i(u, \theta) = u_i \sum_k u_k \left(G_{ik} \sin\theta_{ik} - B_{ik} \cos\theta_{ik}\right). \tag{5}$$

Collecting the unknowns of Table 1 into one vector $x = [\,\theta_i\ (i \neq \text{slack})\,;\ u_i\ (i \in \text{PQ})\,]$, feasibility means driving the *mismatch* between the scheduled power and the injections of (4)–(5) to zero:

$$F(x) = \begin{bmatrix} p_i^{\text{spec}} - P_i(u, \theta), & i \neq \text{slack} \\ q_i^{\text{spec}} - Q_i(u, \theta), & i \in \text{PQ} \end{bmatrix} = 0, \qquad \begin{aligned} p_i^{\text{spec}} &= \hat{p}_{g,i} - p_{d,i}, \\ q_i^{\text{spec}} &= -\,q_{d,i}. \end{aligned} \tag{6}$$

Reactive rows appear only at load buses, because a generator's reactive output is itself free and absorbs any local imbalance. On IEEE 14 this gives 22 equations in 22 unknowns; on IEEE 118 it gives 181 in 181. Equations (4)–(6) follow the standard formulation of the power-flow problem [16].

The Newton iteration. System (6) is nonlinear, so we solve it iteratively. Newton's method replaces F by its local linear approximation and steps to the point where that approximation vanishes. Writing the power-flow Jacobian of the injected powers as $J = \partial(P, Q)/\partial(\theta, u)$, so that $\partial F/\partial x = -J$, the update is

$$J(x_k)\,\Delta x = F(x_k), \qquad x_{k+1} = x_k + \Delta x. \tag{7}$$

Each iteration of (7) therefore costs one evaluation of the mismatch and one sparse linear solve, both far cheaper than an optimization step.

The reason this is fast in our pipeline is the quality of the starting point. Near a solution, the error $e_k = x_k - x^\star$ of Newton's method obeys

$$\|e_{k+1}\| = O\left(\|e_k\|^2\right), \tag{8}$$

which is *quadratic* convergence: the number of correct digits roughly doubles at every step. A cold start would not benefit from this, but the GNN prediction already lies inside the region where (8) holds. In practice, two to three iterations take the worst bus mismatch from several megawatts (about 6 MW on IEEE 14 and 27 MW on IEEE 118) to below 0.01 MW, which matches a classical solver at a small fraction of its cost. We cap the iteration at 20 steps with a tolerance of 10^{-8} p.u.; across every evaluated instance on all three systems the restoration converged well inside that cap, so no fallback path was exercised.

Clipping the set-points. One physical detail matters. A regression model can predict a generator output slightly outside its capability, for example a small non-zero value at a unit whose $p^{\max} = 0$. Before forming p_i^{spec} in (6), we therefore clip every predicted set-point to its limits,

$$\hat{p}_{g,i} \leftarrow \min\big(\max(\hat{p}_{g,i},\, p_{g,i}^{\min}),\, p_{g,i}^{\max}\big). \tag{9}$$

We clip the predicted voltage set-points at PV buses to their permitted range in the same way. Without this step the slack generator would have to absorb the impossible injections, which distorts the dispatch. With it, the restored operating point is both feasible and closer to the optimal dispatch, as Section 4.2 shows.

3.6 Evaluation procedure

Data. We use OPFData [17], a large public benchmark of solved AC-OPF instances derived from PGLib base networks. It provides on the order of 3×10^5 perturbed operating points per system, with labels computed by an interior-point solver. We use the standard 270k/15k/15k train/validation/test split. Before training, we cross-check every instance so that its labels satisfy the AC equations to about 10^{-6} p.u.

Systems. We evaluate on the IEEE 14-, 30-, and 118-bus systems. The 118-bus system reproduces the original study at scale, and the 14- and 30-bus systems extend it to smaller networks. Figure 2 shows the IEEE 14-bus system as a representative example.

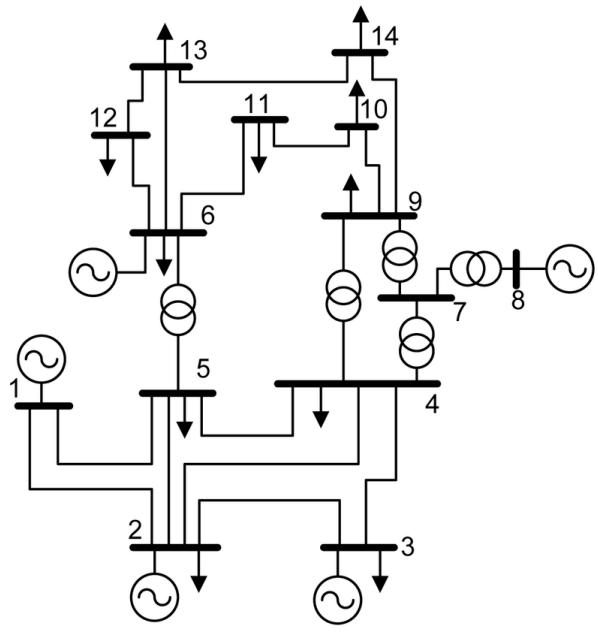


Figure 2 Single-line diagram of the IEEE 14-bus system.

Tools. The DC and AC solutions and the restoration power flow use the PYPOWER package, a Python port of MATPOWER [18]. The reference solver times use PowerModels.jl [19] with the IPOPT interior-point solver. We implement the GNN and the baselines in PyTorch [20] and PyTorch Geometric, and we train them with Adam (learning rate 10^{-3}, batch size 64, K=3 layers, 128 hidden units, and 4 attention heads), gradient clipping, and early stopping. All reported runtimes are median per-instance times; the GNN and the restoration are measured single-threaded on CPU, so they reflect latency rather than batched throughput.

Baselines and metrics. We compare against DC-OPF (the uncorrected baseline) and two standard graph-network models: a graph-convolutional network (GCN, "Model A") and a graph-attention network (GAT, "Model B"). We report three quantities. The *voltage MSE* is the mean squared error over the voltage magnitude and angle at every bus, which is the quantity the original work calls bus voltage. The *generator-power MSE* is the mean squared error over the active and reactive power of every generator, which the original work calls bus power. The *feasibility distance* is the average ℓ_2 norm of the AC power-flow residuals across buses. Where we instead report errors per operating point, we use absolute errors in physical units and label them as such. We also report the per-instance solve time.

4 Results

We evaluate three questions. First, how much does residual learning improve accuracy over standard graph-network baselines, and how does the benefit vary with system size? Second, does the Newton-Raphson restoration deliver feasible operating points? Third, how fast is the full pipeline relative to a classical AC-OPF solver? Throughout, we use three names. The *residual GNN* is the local-attention model of Za'ter et al. [6]. The *Reference* is its raw output, which reproduces that method. The *Proposed* method is our full pipeline: the residual GNN followed by Newton-Raphson restoration. All accuracy figures are test-set mean squared errors. All restoration and runtime figures are averaged over 100 random operating points.

4.1 Residual learning across systems

Residual learning improves every architecture on every system. Table 2 reports the residual-trained models as *raw network outputs*: the restoration layer of Section 3.5 is not applied anywhere in this subsection, so the comparison isolates the effect of the architecture. Restoration is added in Section 4.2.

The local-attention residual GNN attains the lowest bus-voltage error on all three systems, reaching 3.5×10^{-6} on IEEE 14, about 3× below the GCN and GAT baselines. Its margin widens with system size: on IEEE 118 it cuts the bus-voltage MSE to 3.5×10^{-4}, about 10× below the baselines' 3.4×10^{-3}.

Two comparisons with the original work anchor this result. The residual GNN itself reproduces the published figure closely: our 3.48×10^{-4} on IEEE 118 sits within 12% of the 3.12×10^{-4} reported by Za'ter et al. [6], which is the strongest evidence that our independent implementation matches theirs. Our baselines, by contrast, are weaker than the ones they report (3.4×10^{-3} against $\sim 5 \times 10^{-4}$), most likely because our tuning budget was limited to CPU training. The 10× margin above therefore overstates the architectural advantage, and the reproduction claim rests on the absolute agreement of the residual GNN rather than on its margin over our baselines.

Two conventions are worth stating, because Table 2 and Figure 3 report the same models on different scales. The table follows the original work and reports *squared* errors over the full test set, with voltage magnitude and angle combined into a single figure. The heat maps instead report *absolute* errors in physical units ($|\Delta V|$ in per unit, $|\Delta\theta|$ in degrees, $|\Delta P_g|$ in megawatts) over 100 operating points. A voltage MSE of 3.5×10^{-6} therefore corresponds to a mean $|\Delta V|$ of about 2×10^{-3} p.u., not to a different result. Only the feasibility column uses one definition in both places, which is why its values coincide.

The other two columns qualify that result. On IEEE 118 the residual GNN is best on all three metrics. On the two small systems, however, it buys its voltage accuracy at a price: its generator-power error is 4 to 6× above the best baseline, and its raw feasibility is 1.6 to 2.5× worse. The architecture concentrates its capacity on the voltage field, which is what typed attention is

designed to resolve, and a 14- or 30-bus grid offers little topological structure to exploit elsewhere. This also motivates the next subsection directly, because the quantity the architecture sacrifices on small systems is precisely the one that restoration repairs by construction.

Table 2 Residual-trained models on the full test set, as raw outputs before restoration. The *Residual GNN* rows are the same model reported as *Reference* in Table 4. Voltage and power are mean squared errors; feasibility is the average ℓ_2 power-flow residual. Best voltage per system in bold.

System	Model	Voltage MSE	Gen. power MSE	Feasibility (p.u.)
IEEE 14	GCN (A)	1.08×10^{-5}	1.27×10^{-5}	8.06×10^{-3}
	GAT (B)	1.44×10^{-5}	1.16×10^{-5}	9.58×10^{-3}
	Residual GNN	$\mathbf{3.49 \times 10^{-6}}$	5.11×10^{-5}	2.05×10^{-2}
IEEE 30	GCN (A)	2.08×10^{-5}	1.84×10^{-5}	8.72×10^{-3}
	GAT (B)	2.72×10^{-5}	1.59×10^{-5}	9.89×10^{-3}
	Residual GNN	$\mathbf{5.03 \times 10^{-6}}$	9.14×10^{-5}	1.39×10^{-2}
IEEE 118	GCN (A)	3.39×10^{-3}	5.05×10^{-3}	5.92×10^{-2}
	GAT (B)	3.45×10^{-3}	5.12×10^{-3}	5.90×10^{-2}
	Residual GNN	$\mathbf{3.48 \times 10^{-4}}$	3.47×10^{-3}	4.49×10^{-2}

The benefit of residual learning is far larger on the smaller systems. Table 3 reports the voltage-MSE improvement factor between the non-residual and residual variants of each model. Across the three architectures the factors reach 34× on IEEE 30 and 24× on IEEE 14 (for our own model, 25× and 17×), well above the 1.5 to 1.8× (a 35 to 45% reduction) reported in the original work [6]. On the small systems the residual model attains a near-perfect fit ($\sim 10^{-6}$) that the non-residual model cannot, so the ratio is large.

The one exception is instructive. On IEEE 118 the two baselines still gain 4.7×, but the residual GNN gains only 1.1×, because its non-residual variant is already strong (3.9×10^{-4} against 3.5×10^{-4}). Residual learning and a topology-aware architecture are therefore partly substitutable: each supplies the physical grounding that the other would otherwise have to learn, and the benefit of adding the second one is smaller once the first is present.

Table 3 Voltage-MSE improvement from residual learning (non-residual / residual).

System	GCN (A)	GAT (B)	Residual GNN
IEEE 14 (extension)	23.7×	15.1×	17.2×
IEEE 30 (extension)	34.4×	22.1×	25.3×
IEEE 118	4.7×	4.7×	1.1×

4.2 Feasibility restoration

The Newton-Raphson layer of Section 3.5 restores solver-grade feasibility on every system. Table 4 traces three quantities through the pipeline: the feasibility distance, the dispatch error, and the cost error. The dispatch error is the mean absolute error of the generator active powers against the AC-OPF optimum, and the cost error is the relative difference between the generation cost and the AC-OPF optimal cost.

Feasibility improves by five orders of magnitude relative to the DC baseline, and by four relative to the raw network output. The uncorrected DC solution is grossly infeasible ($\sim 10^{-1}$ p.u.),

the raw residual GNN reduces this to $\sim 10^{-2}$ p.u., and the restoration layer drives it to $\sim 10^{-6}$ p.u. This matches the AC-OPF solver to within measurement precision: on all three systems the restored point in fact sits marginally below the solver's own residual (9.95×10^{-7} against 1.11×10^{-6} on IEEE 14, 1.15×10^{-6} against 1.21×10^{-6} on IEEE 30, and 3.16×10^{-6} against 3.57×10^{-6} on IEEE 118), so the metric has reached its numerical floor rather than a limit of the method.

Restoration also improves the two economic quantities, which the original work did not report. Because we clip the predicted set-points to generator limits before the solve (9), the dispatch error falls on every system, for example from 0.16 to 0.09 MW on IEEE 30. The cost error falls with it, roughly halving on IEEE 14 and IEEE 30 (0.19% to 0.09% and 0.11% to 0.05%) and dropping by an order of magnitude on IEEE 118 (0.60% to 0.06%). The restored operating point is therefore not only feasible but also closer to the economic optimum than the raw prediction.

Figure 3 shows the full picture across the four state metrics, including one quantity that restoration does not improve. Restoration wins on bus power and feasibility on every system, and on bus angle for IEEE 30 and IEEE 118. On the two small systems, however, the bus-voltage error grows by about 40%. The cause is structural rather than numerical: the power flow holds the generator-bus voltages at their predicted values and re-derives only the load-bus voltages. On IEEE 14 and IEEE 30 the network predicts load-bus voltages more accurately than generator-bus voltages, so re-deriving the former from the latter passes some of that error on. On IEEE 118 the two groups are predicted about equally well and the effect disappears. The trade is strongly favourable in any case, since a small voltage penalty buys four orders of magnitude of feasibility together with a better and cheaper dispatch.

Table 4 Feasibility restoration over 100 random operating points.

System	Method	Feasibility (p.u.)	Dispatch error (MW)	Cost error (%)
IEEE 14	DC-OPF	2.37×10^{-1}	3.27	5.87
	Reference (GNN raw)	2.07×10^{-2}	0.20	0.19
	Proposed	$\mathbf{9.95\times10^{-7}}$	**0.02**	**0.09**
	AC-OPF	1.11×10^{-6}	0.00	0.00
IEEE 30	DC-OPF	1.44×10^{-1}	2.45	8.65
	Reference (GNN raw)	1.38×10^{-2}	0.16	0.11
	Proposed	$\mathbf{1.15\times10^{-6}}$	**0.09**	**0.05**
	AC-OPF	1.21×10^{-6}	0.00	0.00
IEEE 118	DC-OPF	3.87×10^{-1}	8.59	4.19
	Reference (GNN raw)	4.49×10^{-2}	1.37	0.60
	Proposed	$\mathbf{3.16\times10^{-6}}$	**0.71**	**0.06**
	AC-OPF	3.57×10^{-6}	0.00	0.00

4.3 Runtime

Table 5 compares the proposed pipeline against a classical AC-OPF solve. The pipeline runs three steps in sequence, namely the DC-OPF baseline, GNN inference, and Newton-Raphson restoration, and their sum is the time an operator would actually wait. That total is smaller than the AC-OPF solve on every system, and the advantage grows with size: 1.6× on IEEE 14, 1.9× on IEEE 30, and 3.3× on IEEE 118. The cost of an AC-OPF solve grows far more steeply than GNN inference and a warm-started power flow, so the speedup should widen further on larger grids.



(a) IEEE 14 **(b)** IEEE 30

(c) IEEE 118

Figure 3 Per-metric performance over 100 random operating points. Colour is per-column log-normalized; paler is better.

One point of comparison is worth stating explicitly. Our AC-OPF reference solves IEEE 118 in 129 ms, whereas the original work reports 2467 ms for the same system using a different solver stack. Our 3.3× is therefore measured against a reference roughly 20× faster than theirs, and the two speedup figures are not directly comparable. The absolute pipeline latency, 39.3 ms on IEEE 118, is the quantity an operator would care about.

Table 5 Median per-instance runtime (ms). Total is the complete proposed pipeline; DC- and AC-OPF use the IPOPT interior-point solver.

	Proposed pipeline (ms)				AC-OPF solver	
System	DC-OPF	GNN	Restoration	**Total**	**(ms)**	**Speedup**
IEEE 14	2.5	3.6	4.0	**10.1**	16.5	1.6×
IEEE 30	4.0	4.0	3.9	**11.9**	23.0	1.9×
IEEE 118	23.0	7.3	9.0	**39.3**	129.0	3.3×

Figure 4 summarizes the trade-off that these results describe. DC-OPF is fast but infeasible, AC-OPF is feasible but slow, and the proposed pipeline occupies the fast-and-feasible corner that

neither classical method reaches alone.

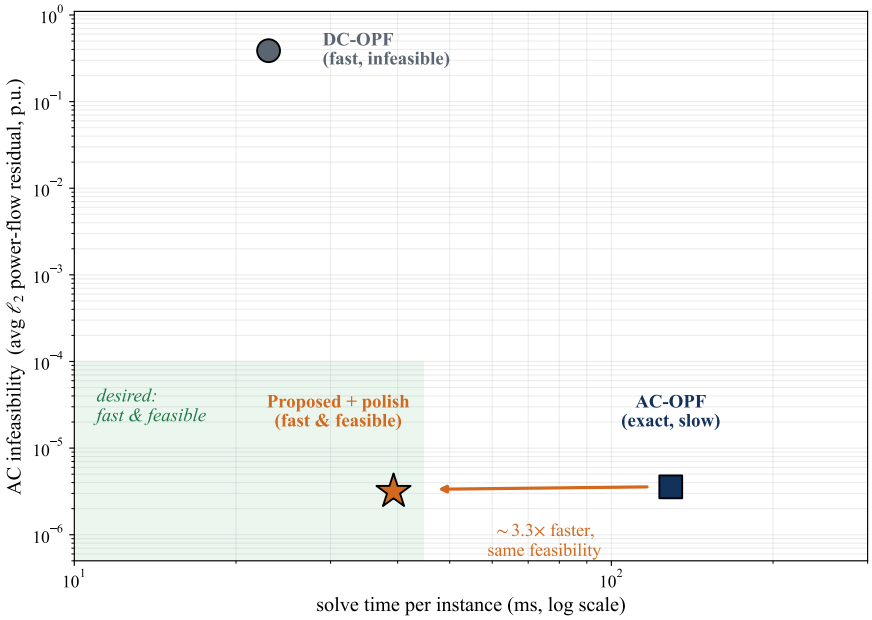


Figure 4 Trade-off between speed and feasibility on IEEE 118.

5 Discussion

The results support three interpretations that together explain when residual learning and feasibility restoration help, and by how much.

Residual learning helps most where the DC baseline is a small, clean correction. Table 3 shows the improvement factor rising as the system shrinks, peaking at 34× on IEEE 30 across the three architectures (25× for our own model). The mechanism is a matter of achievable accuracy rather than architecture. On a small network the residual to be learned is compact, so the residual model attains a near-perfect fit ($\sim 10^{-6}$) that a non-residual model, which must learn the full mapping from scratch, cannot approach; the ratio between the two is therefore large. On IEEE 118 the baselines are further from perfect, so their ratio compresses to ~5×, and the residual GNN gains least of all (1.1×) because its non-residual variant is already strong. This finding extends the original study, which reported gains of 35% to 45% on larger systems [6], and shows that the residual paradigm is even more effective on the smaller networks it did not test.

The proposed architecture's advantage grows with system size. Table 2 shows the residual GNN dominating on bus voltage across all systems, but its margin over the GCN and GAT baselines widens with size, from a factor of a few on IEEE 14 to an order of magnitude on IEEE 118. Typed attention and two-level DC integration exist to capture complex topological coupling, and a 14-bus grid simply contains little of it to exploit. The architecture therefore pays off precisely where the problem is hard, which is the regime that matters for real systems.

Restoration converts a good decision into an exact state. The value of the Newton-Raphson layer is visible in Table 4 and Figure 3. The raw GNN already makes a near-optimal *decision*, since its generator dispatch is within a fraction of a percent of the optimum, but its *state* is only approximately physical. A warm-started power flow re-derives the exact state that this

decision produces, collapsing the feasibility distance by four orders of magnitude for a few extra milliseconds.

The economic side of this step is worth emphasizing, because the original work did not examine it. One might expect a feasibility correction to cost accuracy, trading a cheap but invalid point for a valid but more expensive one. We observe the opposite. Clipping the set-points to generator limits removes injections that a saturated reference bus would otherwise have to absorb, so restoration lowers both the dispatch error and the cost error on every system tested. The cost gap to the AC-OPF optimum roughly halves on IEEE 14 and IEEE 30 and falls by an order of magnitude on IEEE 118, to 0.06%. Feasibility and economics improve together, which makes the step a cheap and principled bridge from a learned approximation to a solver-grade operating point.

Comparison with the original claims. Our reproduction meets or exceeds the paper's accuracy claims but diverges on feasibility, in an informative way. The MSE reductions we measure (an order of magnitude on IEEE 118) surpass the abstract's "up to 25%" figure, although as noted in Section 4.2 that margin partly reflects our weaker baselines; the firmer comparison is the residual GNN's agreement with the published value to within 12%. The paper's claimed 3× feasibility advantage of the proposed architecture over the baselines, however, did not reproduce for the raw network outputs. On IEEE 118 the residual GNN is only about 1.3× more feasible than the baselines, and on IEEE 14 and IEEE 30 it is 1.6 to 2.5× *less* feasible. Part of this divergence may be definitional: the original work reports a maximum absolute violation, whereas we report an average ℓ_2 residual, so the two feasibility columns are not measured on the same scale. Even allowing for that, architecture alone does not deliver feasibility, which is precisely what our restoration layer addresses: instead of relying on the network to be feasible, we make it feasible by construction, overshooting the 3× target by four orders of magnitude. The runtime claim is likewise not directly comparable, since our AC-OPF reference solves IEEE 118 in 129 ms against the 2467 ms reported in the original work. The reproduction thus both verifies the parts of the framework that hold and motivates the extension that fills the part that does not.

6 Broader Impact: Saudi Arabia and Vision 2030

Saudi Arabia is undertaking one of the fastest grid transformations in the world, and the problem studied in this report sits at its centre. Vision 2030 [21] commits the Kingdom to drawing about half of its electricity from renewable sources by 2030 [22, 23], to displacing the liquid fuel still burned in power plants with gas and renewables [22], and to building new load centres such as NEOM around variable generation. Each of these goals makes real-time operation harder in exactly the way the method studied here addresses.

The technical reason is specific to the Kingdom's geography. Large solar and wind plants are sited where the resource is best, which is often far from the coastal load centres, so power crosses long transmission distances whose losses, voltage profile, and reactive-power balance shift with irradiance and wind. These are precisely the quantities the DC approximation discards. The fast tool that control centres rely on is therefore least trustworthy when the system is most stressed, such as a summer afternoon when air-conditioning and desalination push demand toward its annual peak while solar output moves with passing cloud. An operating point that is AC-feasible by construction and returned in tens of milliseconds allows a control centre to re-evaluate its dispatch as often as the resource actually changes, rather than once per scheduling interval. The same speed makes it practical to screen many contingencies or renewable scenarios within a single decision window, which is the analysis that a high-renewable grid needs most.

The economic dimension matters at national scale. On the largest system we tested, the restored solutions sit within 0.06% of the AC-OPF optimum (Table 4) while remaining physically valid, so an operator gives up essentially nothing in cost by adopting the fast method. That is the practical precondition for trusting a learned tool in the first place. Even a small percentage of avoided fuel burn and better use of existing transmission compounds into meaningful savings and deferred capital spending on a national system, which supports the Vision 2030 objectives of economic diversification and of freeing hydrocarbons for higher-value export rather than domestic combustion.

Finally, the approach fits the Kingdom's existing institutional setting. It needs no new hardware, because it wraps around the DC-OPF and power-flow tools that operators already run, and it is trained offline on solved cases that a utility already owns. This makes it deployable on the national grid and on the interconnections that link Saudi Arabia to its neighbours, where fast and physically valid dispatch supports cross-border exchange. Developing the capability locally also builds domestic expertise in grid-scale machine learning, which is itself one of the human-capital aims of Vision 2030.

7 Limitations and Future Work

We state the study's limitations honestly and pair each with a concrete next step.

Compute budget and scale. We trained on CPU rather than the GPU hardware of the original work, which capped the largest system we could reproduce at scale. The paper's 2000-bus case would require roughly two orders of magnitude more data storage and training time than our setup allows, so we omitted it. A GPU run would restore the full-scale 2000-bus experiment and sharpen the runtime comparison, where the speedup is expected to reach an order of magnitude.

Restoration enforces equalities, not all inequalities. The Newton-Raphson layer drives the power-balance *equalities* to solver tolerance, and it respects generator active-power limits through the clipping step. It does not, however, enforce every operating *inequality* (line thermal ratings or reactive-power ranges), so a small residual limit violation can remain. Enforcing generator reactive limits inside the power flow, or following the restoration with one projection step, would close this gap while preserving the millisecond cost.

Restoration can shift error onto the load-bus voltages. Because the power flow holds the predicted generator-bus voltages fixed and re-derives the load-bus voltages from them, any error in the former is passed to the latter. On IEEE 14 and IEEE 30, where the network already predicts load-bus voltages more accurately than generator-bus voltages, this raises the bus-voltage error by about 40% even as feasibility, dispatch, and cost all improve. Weighting generator-bus voltages more heavily during training, or letting the restoration adjust them within their limits, would remove this side effect.

Two of the original case studies were not reproduced. The original study also evaluated robustness under $N-1$ contingencies and the use of the trained model as a data generator for new training samples. We reproduced neither, because both require the topology-perturbed variants of OPFData and a second round of training that our compute budget did not allow. Our reproduction claim therefore covers the accuracy and residual-learning results, not the full set of case studies. Running the $N-1$ variants would also test whether the restoration layer holds up when a line or generator is lost, which is the setting where feasibility matters most.

Single-period, fixed-topology scope. Our study addresses static, single-period OPF on a fixed topology. Real operations are sequential and face contingencies. Extending the residual target to time-series and multi-period OPF, and training across a distribution of topologies, would test the approach under the conditions that matter for dispatch and security assessment.

Evaluation sample for restoration and runtime. The restoration and runtime results are averaged over 100 random operating points rather than the full 15k test split, because each point requires a separate power-flow solve and a serial reference AC-OPF solve for comparison. The accuracy results in Table 2 use the complete test set. A larger restoration sample would tighten the runtime medians, although the feasibility figures are already at the numerical floor of the metric.

The dispatch gap narrows but does not close. Restoration improves the dispatch on every system, yet a gap remains: the mean generator error is still 0.71 MW on IEEE 118, the largest of the three systems. Part of the DC-to-AC correction is close to discontinuous, because the AC optimum may commit a generator for voltage support using information the DC solution does not contain, and a smooth regressor cannot reproduce such a switch. Feeding the network richer reactive-adequacy features, or letting the restoration perform a limited economic re-dispatch, are promising ways to close the remainder.

8 Conclusion

We reproduced a residual-learning framework for AC optimal power flow and extended it into a pipeline that returns solver-grade, AC-feasible operating points at near-DC speed. Reproducing the topology-aware residual GNN, we confirmed that learning the DC-to-AC correction cuts the bus-voltage error by an order of magnitude over graph-convolutional and graph-attention baselines on IEEE 118, and we showed on the previously untested IEEE 14 and 30 systems that the benefit is even larger, reaching 34×, because the residual model attains a near-perfect fit on smaller networks. Our main contribution is a warm-started Newton-Raphson feasibility-restoration layer. It drives the average AC power-flow residual from $\sim 10^{-2}$ to $\sim 10^{-6}$ p.u., matching a classical interior-point solver while running about 3.3× faster on IEEE 118. Because it clips the predicted set-points to generator limits, it also improves the economics of the solution, lowering the dispatch error on every system and cutting the cost error by roughly half on IEEE 14 and IEEE 30 and by an order of magnitude on IEEE 118. Physics-restored residual learning therefore bridges the long-standing gap between fast DC-OPF and exact AC-OPF, offering a practical path to near-real-time, AC-feasible grid operation as power systems absorb ever more renewable and distributed generation.

References

[1] Stephen Frank and Steffen Rebennack. “An introduction to optimal power flow: Theory, formulation, and examples”. In: *IIE Transactions* 48.12 (2016), pp. 1172–1197.

[2] Ahmed Babiker et al. “Optimal Power Flow: A Review of State-of-the-Art Techniques and Future Perspectives”. In: *IEEE Access* 13 (2025), pp. 60012–60039.

[3] Juan Caballero-Peña et al. “Distributed energy resources on distribution networks: A systematic review of modelling, simulation, metrics, and impacts”. In: *International Journal of Electrical Power & Energy Systems* 138 (2022), p. 107900.

[4] Han Pyo Lee. "Integrating distributed energy resources in power distribution systems: A comprehensive review of impacts, challenges, and opportunities from a distribution system operator's perspective". In: *Applied Energy* (2025).

[5] Y. M. Atwa et al. "Optimal Renewable Resources Mix for Distribution System Energy Loss Minimization". In: *IEEE Transactions on Power Systems* 25.1 (2010), pp. 360–370.

[6] Muhy Eddin Za'ter, Bri-Mathias Hodge, and Kyri Baker. "Residual correction models for AC optimal power flow using DC optimal power flow solutions". In: *Electric Power Systems Research* 262 (2027), p. 113612.

[7] Cheng Yang et al. "Optimal Power Flow in Distribution Network: A Review on Problem Formulation and Optimization Methods". In: *Energies* 16.16 (2023), p. 5974.

[8] Hamed Ahmadi and José R. Martí. "Distribution System Optimization Based on a Linear Power-Flow Formulation". In: *IEEE Transactions on Power Delivery* 30.1 (2015), pp. 25–33.

[9] Mohamed A. M. Shaheen et al. "Optimal Power Flow of Power Systems Including Distributed Generation Units Using Sunflower Optimization Algorithm". In: *IEEE Access* 7 (2019), pp. 109289–109300.

[10] Damian Owerko, Fernando Gama, and Alejandro Ribeiro. "Optimal Power Flow Using Graph Neural Networks". In: *ICASSP 2020 – 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 5930–5934. DOI: 10.1109/ICASSP40776.2020.9053140.

[11] Jian Zhang et al. "GAT-OPF: Robust and scalable topology analysis in AC optimal power flow with graph attention networks". In: *IET Generation, Transmission & Distribution* 19.1 (2025), e70039.

[12] Hooman Khaloie et al. "Review of machine learning techniques for optimal power flow". In: *Applied Energy* 388 (2025), p. 125637.

[13] Priya L. Donti, David Rolnick, and J. Zico Kolter. "DC3: A Learning Method for Optimization with Hard Constraints". In: *International Conference on Learning Representations (ICLR)*. 2021.

[14] Kyri Baker. "Learning Warm-Start Points for AC Optimal Power Flow". In: *2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2019, pp. 1–6. DOI: 10.1109/MLSP.2019.8918690.

[15] Frederik Diehl. "Warm-Starting AC Optimal Power Flow with Graph Neural Networks". In: *NeurIPS Workshop on Tackling Climate Change with Machine Learning*. 2019.

[16] John J. Grainger and William D. Stevenson. *Power System Analysis*. McGraw-Hill, 1994.

[17] Sean Lovett et al. *OPFData: Large-scale datasets for AC optimal power flow with topological perturbations*. Tech. rep. Datasets distributed via the public `gridopt-dataset` Google Cloud bucket. Google DeepMind, 2024.

[18] Ray D. Zimmerman, Carlos E. Murillo-Sánchez, and Robert J. Thomas. "MATPOWER: Steady-State Operations, Planning, and Analysis Tools for Power Systems Research and Education". In: *IEEE Transactions on Power Systems* 26.1 (2010), pp. 12–19.

[19] Carleton Coffrin et al. "PowerModels.jl: An Open-Source Framework for Exploring Power Flow Formulations". In: *2018 Power Systems Computation Conference (PSCC)*. 2018, pp. 1–8. DOI: 10.23919/PSCC.2018.8442948.

[20] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 32. 2019.

[21] Kingdom of Saudi Arabia. *Saudi Vision 2030*. https://www.vision2030.gov.sa/. Accessed 2026. 2016.

[22] Ministry of Energy, Kingdom of Saudi Arabia. *National Renewable Energy Program*. https://www.moenergy.gov.sa/en/eco-system/programs/renewable-energy. Accessed 2026.

[23] Saudi Green Initiative. *Saudi Green Initiative: Reducing Carbon Emissions*. https://www.sgi.gov.sa/about-sgi/sgi-targets/reduce-carbon-emissions/. Accessed 2026.