Alternative Learning Architecture for Solving AC-OPF via Supervised Relaxation and Cross Encoder

Hien Thanh Doan
Department of Electronic Engineering
Sogang University
Seoul 04107, South Korea
hiendoanht@sogang.ac.kr

Keunju Song
Department of Electronic Engineering
Sogang University
Seoul 04107, South Korea
kjsong4089@sogang.ac.kr

Kibaek Kim
Mathematics and Computer Science Division
Argonne National Laboratory
Lemont, Illinois, USA
kimk@anl.gov

Hongseok Kim
Department of Electronic Engineering
Sogang University
Seoul 04107, South Korea
hongseok@sogang.ac.kr

Abstract

As power systems evolve, efficient AC optimal power flow solutions are increasingly critical, yet traditional methods face challenges in speed and scalability. This paper introduces ExpressOPF, an ML-based framework that alternates between the original AC-OPF problem and its relaxed form. By integrating Lagrange multipliers into a compact neural network, ExpressOPF enforces constraints while improving accuracy and inference speed. Tested on 162-, 300-, 1354-, and the real-world 4492-bus system operated by Korea Power Exchange (KPX), ExpressOPF achieves under 1% cost deviation from MATPOWER, 100,000$\times$ speedup, 75% model compression, and 40% lower GPU memory use—while maintaining over 99% constraint satisfaction. These results highlight its potential for real-time, resource-efficient AC-OPF at scale, with future work aimed at grid reliability and multi-area operations.

1 Introduction

The Alternating Current Optimal Power Flow (AC-OPF) problem lies at the core of secure and economic power system operation, aiming to optimize generation dispatch while satisfying nonlinear physical and engineering constraints. As power grids scale up and integrate more renewable energy, the need for fast and scalable AC-OPF solvers becomes increasingly critical—especially for real-time applications. Traditional solvers, such as nonlinear programming (NLP) and interior-point methods, have been the mainstay of OPF computation but struggle with the non-convexity and dimensionality of modern networks. Recent efforts to accelerate numerical methods using multi-core CPUs and GPUs have shown promise [1, 2], but face challenges in memory management and ensuring feasibility, particularly under real-time constraints.

To address these limitations, machine learning (ML) has emerged as a viable alternative. Supervised learning (SL) approaches train deep neural networks (DNN) to directly map system states to OPF solutions using labeled datasets [3, 4]. While SL models can achieve high accuracy, they suffer from generalization issues and require extensive labeled data, which is costly to generate for large-scale systems. In contrast, unsupervised learning (UL) bypasses the need for labels by embedding power

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: GPU-Accelerated and Scalable Optimization (ScaleOPT).

flow physics directly into the training process [5, 6]. However, UL may struggle to enforce feasibility and may yield locally optimal solutions.

To bridge this gap, semi-supervised learning (SSL) offers a hybrid paradigm that leverages both labeled and unlabeled data [7, 8]. Although SSL has demonstrated potential in various power system tasks—including transient stability assessment and event classification [9]—it remains underexplored for AC-OPF. Recent SSL-based methods incorporate physical priors or post-processing techniques, but often lack flexibility and efficiency due to their reliance on direct mappings and limited problem formulations. In this work, we present a GPU-accelerated, semi-supervised learning framework for AC-OPF that alternates between supervised and unsupervised objectives. By integrating relaxed problem formulations and compact neural network (NN) architectures, our approach improves scalability, generalization, and training efficiency—demonstrating strong performance across benchmark power systems.

The main contributions of this paper can be summarized as follows: 1) We propose ExpressOPF, a novel ML-based framework using alternating learning with Lagrange multipliers to switch between the original and relaxed AC-OPF problems. This enhances convergence and enables a compact DNN suitable for SSL. 2) The learning is cast as an augmented Lagrangian optimization within a semi-supervised setup, where both problems are solved iteratively and independently. Alternating updates of penalty weights allow mutual reinforcement to improve feasibility and convergence. 3) Experiments on 162-, 300-, 1354-, and Korea 4492-bus systems show that ExpressOPF outperforms six state-of-the-art methods, reducing model size by 75%, GPU memory by 40%, achieving >99% constraint satisfaction with <1% cost gap to MATPOWER, and up to 100,000× faster inference ($20\mu s$ for Korea 4492-bus systems).

2 AC-OPF Problem

We represent the power system as an undirected connected graph $\mathcal{G} = (\mathcal{I}, \mathcal{L})$, where $\mathcal{I}$ denotes the set of buses and $\mathcal{L}$ the set of branches (transmission lines or transformers). Each bus $i \in \mathcal{I}$ may be connected to generators and/or loads, and each branch $(i, j) \in \mathcal{L}$ has an apparent power limit $\overline{S}_{ij}$. Let y denote the vector of decision variables, including the active power generation $P_i^g \in [\underline{P}_i^g, \overline{P}_i^g]$, reactive power generation $Q_i^g \in [\underline{Q}_i^g, \overline{Q}_i^g]$, branch apparent power flow $S_{ij} \in [\underline{S}_{ij}, \overline{S}_{ij}]$, voltage angle $\theta_i \in [\underline{\theta}_i, \overline{\theta}_i]$, voltage angle difference $\theta_{ij} \in [\underline{\theta}_{ij}, \overline{\theta}_{ij}]$, and voltage magnitude $V_i \in [\underline{V}_i, \overline{V}_i]$. The system inputs are denoted by $x = \{P_i^d, Q_i^d\}_{i \in \mathcal{I}}$, representing the active and reactive power demands at each bus. The nonlinear AC-OPF problem is formulated in **P1** as follows:

$$\underset{y \in \mathbb{R}^n}{\textbf{minimize}} \quad f_x(y) \tag{1a}$$

$$\textbf{subject to} \quad g(y) \leq 0, h(y) = 0, \tag{1b}$$

where $f_x(y)$ is a convex (typically quadratic) generation cost function parameterized by demand x; $g(y) \leq 0$ includes inequality constraints such as bounds on generation, voltages, and line flows. $h(y) = 0$ enforces the nonlinear AC power flow equations and nodal power balance.

3 Alternative Learning Framework

Proposed Cross Encoder Architecture for Unsupervised Learning: In solving AC-OPF for UL, we apply the Lagrangian method, which is traditionally used to solve the AC-OPF problem via dual variables [18]. Unlike prior work [4, 6] using standard Lagrangian formulations, we adopt an augmented Lagrangian approach [10], which incorporates dual variables and penalties. The loss function for UL of **P1** is the augmented Lagrangian function given by:

$$\mathcal{L}^{\text{UL}} = \sum_{i \in \mathcal{N}_g} C_i(\hat{P}_i^g) + \sum_{\ell \in \mathcal{U}} \left[\lambda^{\text{UL}}(\hat{y}_\ell)\, \text{R}(\hat{y}_\ell) + \frac{\rho}{2} \left\| \text{R}(\hat{y}_\ell) \right\|^2 + \mu^{\text{UL}}(\hat{y}_\ell)\, \text{H}(\hat{y}_\ell) + \frac{\rho}{2} \left\| \text{H}(\hat{y}_\ell) \right\|^2 \right] \tag{2}$$

where penalty parameter $\rho > 0$ is fixed during training and serves as a stabilization constant that penalizes constraint violations, improving conditioning and preventing oscillations [4]. The term $\hat{y}$ denotes the full vector of *all* decision variables predicted by the model, and the index ℓ runs over $\mathcal{U}$, a set of all components of $\hat{y}$, including $\hat{P}_i^g$, $\hat{Q}_i^g$, $\hat{P}_i^d$, $\hat{Q}_i^d$, $\hat{P}_i^0$, $\hat{Q}_i^0$ for all $i \in \mathcal{N}$, and $\hat{\theta}_{ij}$, $\hat{S}_{ij}$ for all

$(i,j) \in \mathcal{E}$. The violation of inequality constraint can be written as:

$$\mathrm{R}(\hat{y}_i) = \max(\hat{y}_i - \overline{y}_i, 0) + \max(\underline{y}_i - \hat{y}_i, 0), \quad \text{for } y_i \in \{P_i^g, Q_i^g, \theta_{ij}\}, \tag{3}$$

$$\mathrm{R}(\hat{S}_{ij}) = \max(\hat{S}_{ij} - \overline{S}_{ij}, 0). \tag{4}$$

Similarly, for equality constraint violations, we define:

$$\mathrm{H}(\hat{y}_i) = |\hat{y}_i - y_i|, \quad \text{for } y_i \in \{P_i^d, Q_i^d, P_i^0, Q_i^0\}. \tag{5}$$

Following [11], the dual variables in the UL task are updated at each epoch by incrementing the multipliers $\lambda^{\mathrm{UL}}(\hat{y}_\ell)$ for inequality constraints and $\mu^{\mathrm{UL}}(\hat{y}_\ell)$ for equality constraints as follow:

$$\lambda^{\mathrm{UL}}(\hat{y}_\ell) \leftarrow \lambda^{\mathrm{UL}}(\hat{y}_\ell) + p^{\mathrm{UL}}\mathrm{R}(\hat{y}_\ell), \tag{6}$$

$$\mu^{\mathrm{UL}}(\hat{y}_\ell) \leftarrow \mu^{\mathrm{UL}}(\hat{y}_\ell) + s^{\mathrm{UL}}\mathrm{H}(\hat{y}_\ell). \tag{7}$$

As the NN model, we employ a cross encoder architecture to address the high-dimensional nature of the AC-OPF problem. Unlike traditional fully connected models [5, 6, 7, 8], our approach introduces a bottleneck encoder that compresses the input demand vector $x = [P^d; Q^d] \in \mathbb{R}^{|\mathcal{N}_P|+|\mathcal{N}_Q|}$ into a compact latent representation $z \in \mathbb{R}^d$, where $d \ll |\mathcal{N}_P| + |\mathcal{N}_Q|$. A separate decoder then reconstructs the output $\hat{o} = [\hat{V}; \hat{\theta}] \in \mathbb{R}^{2|\mathcal{N}|}$, representing the predicted voltage magnitudes $\hat{V}$ and angles $\hat{\theta}$. The proposed cross encoder differs from standard auto-encoder in that the decoder is independent of encoder, allowing greater flexibility in the alternating learning loop. The architecture of NN as follow:

$$h_1 = \text{LeakyReLU}(W_1 x + b_1), \quad z = \text{LeakyReLU}(W_2 h_1 + b_2),$$
$$h_2 = \text{LeakyReLU}(W_3 z + b_3), \quad \hat{o} = \text{Sigmoid}(W_4 h_2 + b_4),$$

where W_k and b_k are trainable weights. Note that the decision variables $\hat{y}$ are reconstructed from $\hat{o}$ using physical laws (e.g., Ohm's and Kirchhoff's), evaluated in parallel on the GPU for efficiency.

Supervised Relaxation Learning: Thermal limit constraints, though crucial for safety, are typically non-binding under normal conditions. While convex in form in **P1**, they introduce non-convexities due to their nonlinear dependence on voltage magnitudes and angles [12]. To improve solver convergence and learning efficiency, we relax these constraints and reformulated as **P2** in our framework. This relaxation simplifies learning, improves generalization, and accelerates convergence by removing rarely active constraints. It also yields a tractable surrogate with a lower-bound cost, enabling optimality gap evaluation. Since thermal limits vary dynamically and labeled data is often sparse [13], relaxing these constraints allows robust learning without exhaustive labels. The model remains valid even if line ratings change, avoiding the need to regenerate labeled data. The resulting relaxed problem **P2**, the loss function of SL task can be formulated as follow,

$$\mathcal{L}^{\mathrm{SL}} = \sum_{i \in \mathcal{N}_g} C_i(\hat{P}_i^g) + \sum_{\ell \in \mathcal{S}} \mu^{\mathrm{SL}}(\hat{y}_\ell)\, \mathrm{K}(\hat{y}_\ell) + \frac{\rho}{2} \left\| \mathrm{K}(\hat{y}_\ell) \right\|^2 \tag{8}$$

where $\hat{y}$ denotes the vector of all decision variables predicted by the model of $\hat{y}$ in the SL task, and the index ℓ runs over $\mathcal{S}$, a set of all components, including $\hat{P}_i^g$, $\hat{Q}_i^g$, $\hat{P}_i^d$, $\hat{Q}_i^d$, $\hat{P}_i^0$, $\hat{Q}_i^0$ for all $i \in \mathcal{N}$, and $\hat{\theta}_{ij}$ for all $(i,j) \in \mathcal{E}$, but *excluding* $\hat{S}_{ij}$. The function $\mathrm{K}(\cdot)$ quantifies how far each predicted variable $\hat{y}_\ell$ deviates from its corresponding true value y_ℓ. It is defined component-wise as:

$$\mathrm{K}(\hat{y}_i) = |\hat{y}_i - y_i|, \quad \text{for } y_i \in \{P_i^g, Q_i^g, P_i^d, Q_i^d, \theta_{ij}\}, \tag{9}$$

$$\mathrm{K}(\hat{y}_i) = |\hat{y}_i|, \quad \text{for } y_i \in \{P_i^0, Q_i^0\}. \tag{10}$$

Hence, $\mathrm{K}(\cdot)$ serves a similar role to $\mathrm{H}(\cdot)$ in (2), but instead of measuring constraint violation, it captures the SL error between predicted and ground truth values. Since **P2** involves the equality constraints, we have the corresponding the dual variable denoted as μ^{SL} and is updated as follow.

$$\mu^{\mathrm{SL}}(\hat{y}_\ell) \leftarrow \mu^{\mathrm{SL}}(\hat{y}_\ell) + s^{\mathrm{SL}}\mathrm{K}(\hat{y}_\ell). \tag{11}$$

Alternative Learning and Convergence of Proposed Method: We propose an alternative learning architecture that jointly trains on **P1** and **P2**. As illustrated in Fig. 1, the two problems differ in both formulation and solution, with a partially overlapping feasible region (gray) and distinct regions exclusive to each (blue for **P1**, orange for **P2**). The training begins with SL on **P2**, using $(x_{\mathrm{SL}}, \hat{o}_{\mathrm{SL}})$ to predict decision variables $\hat{y}_{\ell \in \mathcal{S}}$ and update gradients, followed by UL on **P1**, using $(x_{\mathrm{UL}}, \hat{o}_{\mathrm{UL}})$ to predict $\hat{y}_{\ell \in \mathcal{U}}$, as shown in Fig. 2. Alternating between **P1** and **P2** improves convergence in the shared region but may destabilize learning in non-overlapping regions due to conflicting objectives. To address this, the SL phase on **P2** is terminated once the loss plateaus, ensuring stable optimization focused solely on **P1**. Note that **P1** and **P2** share a common NN.

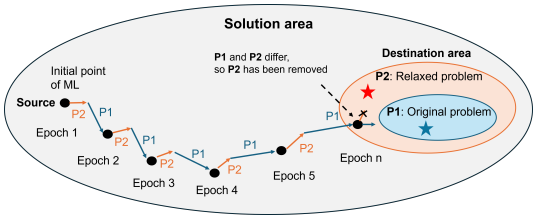

Figure 1: Transitioning from alternative learning to focused learning.

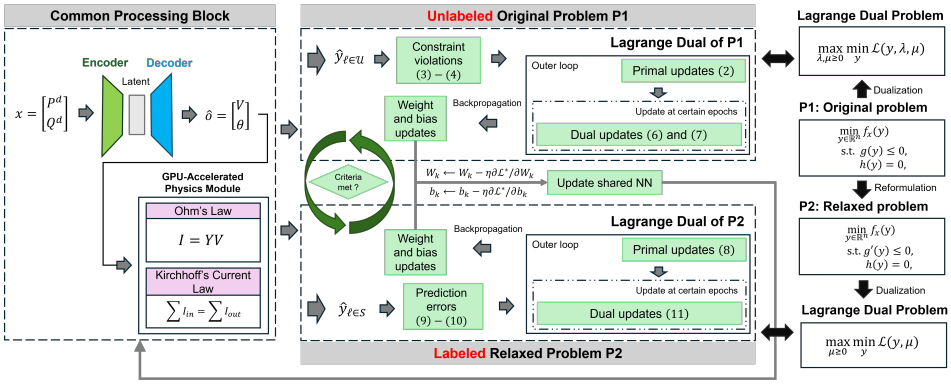

Figure 2: Proposed alternative learning framework for solving the AC-OPF problem using a cross encoder structure, integrating both SL and UL training paths.

4 Numerical Results

We evaluate the proposed method against six state-of-the-art approaches, including mathematical solvers and both SL- and UL-based learning models, across power systems with 162, 300, and 1354 buses, as well as the 4492-bus real-world system of South Korea. These include: **MadSuite** [1], a numerical solver using MadNLP.jl and ExaModels.jl (UMFPACK for CPU and cuDSS for GPU); **DeepOPF-V** [3], an SL model mapping power demands to bus voltages; **LagM** [4], a hybrid SL method integrating Lagrange multipliers; **DC3** [5], an UL approach with an implicit optimization layer; and **DeepLDE** [6], an improved UL model incorporating dual variables.

To ensure a fair comparison, we benchmark all methods using standard performance metrics: **Constraint Satisfaction Ratio**: Measures the percentage of bound constraints satisfied for active power generation P^g_{rate}, reactive power generation Q^g_{rate}, bus voltage limits V_{rate}, branch power flow S_{rate}, and branch angle limits θ_{rate}. **Load Satisfaction Ratio**: Quantifies the percentage of satisfied active P^d_{rate} and reactive Q^d_{rate} power loads across the system. **Cost Difference** ($\mathcal{C}_d$): Average relative deviation between the objective value from MATPOWER and the data-driven model's prediction. **Inference Time** (t_{pred}): Average computation time for solving AC-OPF. **Speedup Factor**: Ratio of MATPOWER's computation time to that of the data-driven method, reflecting computational efficiency. **Zero-Bus Satisfaction Ratio**: Evaluates power balance at zero-injection buses for active (P^0_{rate}) and reactive (Q^0_{rate}) power, computed as:

$$P^0_{\text{rate}} = 100 \times \left(1 - \frac{\sum_{j\in\mathcal{Z}} |\hat{P}_j|}{\sum_{i\in\mathcal{N}} |P_i|}\right)\%, \quad Q^0_{\text{rate}} = 100 \times \left(1 - \frac{\sum_{j\in\mathcal{Z}} |\hat{Q}_j|}{\sum_{i\in\mathcal{N}} |Q_i|}\right) \tag{12}$$

where $\mathcal{Z} \subseteq \mathcal{N}$ is the set of zero-injection buses.

4.1 Comparative Evaluation

Numerical solver baseline: We benchmark ExpressOPF against MadSuite [1], a state-of-the-art AC-OPF solver on both CPU and GPU. As shown in Table 1, MadSuite achieves 99.99% constraint satisfaction and generation costs identical to MATPOWER. ExpressOPF incurs a slightly higher cost (0.1–0.8% above MadSuite) but delivers substantial inference speedups—over 8000× faster—highlighting its practical advantage for large-scale systems.

Table 1: Results on 162- and 300-bus systems.

Case 162-bus system							
Technique	MadNLP+UMFPACK +ExaModels	MadNLP+cuDSS +ExaModels	SL		UL		SSL
Power flow	Equality constraint		Direct mapping method		Implicit layer		Penalty method
Metrics	**MadSuite CPU [1]**	**MadSuite GPU [1]**	**DeepOPF-V [3]**	**LagM [4]**	**DC3 [5]**	**DeepLDE [6]**	**ExpressOPF (proposed)**
P^g_{rate} (%)	100	100	99.93	99.92	100	100	99.99
Q^g_{rate} (%)	100	100	99.87	99.79	99.99	99.99	99.99
S_{rate} (%)	100	100	98.42	97.69	97.93	99.89	99.53
θ_{rate} (%)	100	100	100	100	100	100	100
V_{rate}(%)	100	100	100	100	98.86	98.89	100
$\theta_{ij,\text{rate}}$ (%)	100	100	100	100	100	100	100
P^d_{rate} (%)	99.99	99.99	99.67	99.56	99.99	99.99	99.62
Q^d_{rate} (%)	99.99	99.99	99.63	99.19	99.99	99.99	99.21
P^0_{rate} (%)	99.99	99.99	96.32	95.67	99.99	99.99	99.79
Q^0_{rate} (%)	99.99	99.99	94.4	94.87	99.99	99.99	99.46
$\mathcal{C}_d$ (%)	0.0	0.0	0.4	0.3	4.5	2.1	0.8
t_{pred} (ms) (speedup)	323 (0.3×)	430 (0.2×)	0.143 (699×)	0.14 (714×)	57 (1.8×)	55 (1.7×)	0.012 (8333×)
Case 300-bus system							
P^g_{rate} (%)	100	100	99.7	99.69	x	x	99.97
Q^g_{rate} (%)	100	100	99.71	99.73	x	x	99.94
S_{rate} (%)	100	100	99.53	99.68	x	x	100
θ_{rate} (%)	100	100	100	100	x	x	100
V_{rate} (%)	100	100	100	100	x	x	100
$\theta_{ij,\text{rate}}$ (%)	100	100	100	100	x	x	100
P^d_{rate} (%)	99.99	99.99	97.88	98.42	x	x	99.23
Q^d_{rate} (%)	99.99	99.99	98.3	98.71	x	x	99.29
P^0_{rate} (%)	99.99	99.99	97.55	97.45	x	x	99.92
Q^0_{rate} (%)	99.99	99.99	98.12	98.03	x	x	99.78
$\mathcal{C}_d$ (%)	0.0	0.0	1.3	0.6	x	x	0.1
t_{pred} (ms) (speedup)	329 (0.5×)	568 (0.3×)	0.133 (1278×)	0.133 (1278×)	x	x	0.012 (14166×)

Table 2: Performance result of ExpressOPF on 1354-bus system and South Korea power system.

Case 1354-bus system												
	P^g_{rate}(%)	Q^g_{rate}(%)	S_{rate}(%)	θ_{rate}(%)	V_{rate}(%)	$\theta_{ij,\text{rate}}$(%)	P^d_{rate}(%)	Q^d_{rate}(%)	P^0_{rate}(%)	Q^0_{rate}(%)	$\mathcal{C}_d$(%)	t_{pred} (ms) (speedup)
ExpressOPF	99.86	99.69	99.75	100	100	100	99.66	99.47	99.12	99.54	0.5	0.013($10^5\times$)
South Korea power system (Korea 4492-bus)												
ExpressOPF	99.89	99.97	99.98	100	100	100	99.4	98.51	99.83	99.21	0.9	0.02 ($10^5\times$)

Supervised learning baselines: On the 162-bus system, DeepOPF-V [3] and LagM [4] achieve high overall satisfaction rates ($\sim$99%) with low cost errors, but both struggle with the zero-injection constraint in (12), reaching only 96.32% and 95.67%, respectively. This limitation persists on the 300-bus system, where their zero-injection satisfaction is 97.55% and 97.45%. In contrast, ExpressOPF achieves 99.79% on the 162-bus case and 99.92% on the 300-bus case, significantly improving zero-injection accuracy. These results suggest that while DeepOPF-V and LagM fail to capture zero-injection behavior, ExpressOPF ensures better generalization and stricter constraints.

Unsupervised learning baselines: DC3 [5] and DeepLDE [6] exhibit larger cost gaps of 4.5% and 2.1%, respectively. Despite taking 3× longer to train, DeepLDE achieves only 98.8% voltage satisfaction as illustrated in Table 1, revealing limitations in maintaining feasibility. Moreover, both methods fail on the 300-bus system due to Newton-Raphson divergence, suggesting difficulty escaping local optima in larger networks. In contrast, the proposed method consistently finds better solutions by incorporating limited labeled data, guiding the optimization away from poor local minima and toward globally feasible outcomes.

Scalability and Effectiveness on Large Systems: We evaluate ExpressOPF on the 1354-bus system using a batch size of 256 (Table 2). It satisfies over 99% of constraints with a 0.5% optimality gap. The 13 μs inference time (about 100,000× faster) stems from replacing iterative Newton/IPM steps and repeated Jacobian/KKT factorizations with a single feed-forward pass composed of matrix multiplications that run efficiently and in batch on the GPU. To further examine scalability, we test ExpressOPF on Korea's 4492-bus system operated by KPX. With increased model capacity,

ExpressOPF attains a cost gap of approximately 0.9%, 99.9% satisfaction on inequality constraints, and above 99% on most equality constraints, with the lowest being 98.51% for the reactive power balance Q^d_{rate}. These results demonstrate the robustness and effectiveness of ExpressOPF at scale.

Model compactness and efficiency: To evaluate computational efficiency on the 1354-bus system, we compare our compact learning model (1024–512–1024) with the fully connected DeepOPF-NGT architecture (3072–3072)[7]; both use the same inputs (P_d, Q_d) and outputs (V, θ). Our model reduces parameters from 22M to 4.9M, FLOPs from 50M to 20M, model size from 256 MB to 64 MB, and GPU memory usage from 8.4 GB to 5 GB, while maintaining comparable AC-OPF accuracy with higher efficiency.

5 Conclusion

This study presents ExpressOPF, an ML-based framework for efficient and accurate AC-OPF optimization. It employs an alternating training strategy between the original problem **P1** and its relaxed form **P2**, guided by Lagrange multipliers and compact learning. Once convergence stabilizes, training proceeds solely on **P1** without labeled data. The training pipeline is further refined through convergence and feasibility analysis. Extensive simulations on 162-, 300-, and 1354-bus systems show that ExpressOPF consistently achieves over 99% constraint satisfaction and reduces cost differences to below 1.0% compared to MATPOWER—offering both scalability and improved economic performance over existing methods.

Acknowledgments

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. RS-2025-02215243). This material is based upon work supported by the U.S. Department of Energy, Office of Science, under contract number DE-AC02-06CH11357.

References

[1] S. Shin, M. Anitescu, and F. Pacaud. Accelerating optimal power flow with GPUs: SIMD abstraction of nonlinear programs and condensed-space interior-point methods. *Elect. Power Syst. Res.*, 236:110651, 2024.

[2] Y. Kim and K. Kim. Accelerated computation and tracking of AC optimal power flow solutions using GPUs. In *ICPP Workshops '22*, pages 1–8, 2022.

[3] W. Huang, X. Pan, M. Chen, and S. H. Low. DeepOPF-V: Solving AC-OPF problems efficiently. *IEEE Trans. Power Syst.*, 37(1):800–803, 2021.

[4] F. Fioretto, T. W. Mak, and P. Van Hentenryck. Predicting AC optimal power flows: Combining deep learning and lagrangian dual methods. *Proc. AAAI Conf. Artif. Intell.*, 34(01):630–637, 2020.

[5] L. D. Priya, D. Rolnick, and J. Z. Kolter. DC3: A learning method for optimization with hard constraints. *ICLR*, 2021.

[6] M. Kim and H. Kim. Unsupervised deep lagrange dual with equation embedding for AC optimal power flow. *IEEE Trans. Power Syst.*, 2024.

[7] W. Huang, M. Chen, and S. H. Low. Unsupervised learning for solving ac optimal power flows: Design, analysis, and experiment. *IEEE Trans. Power Syst.*, 2024.

[8] K. Chen, S. Bose, and Y. Zhang. Physics-informed gradient estimation for accelerating deep learning-based AC-OPF. *IEEE Trans. Ind. Informat.*, 2025.

[9] F. Yang, Z. Ling, Y. Zhang, X. He, Q. Ai, and R. C. Qiu. Event detection, localization, and classification based on semi-supervised learning in power grids. *IEEE Trans. Power Syst.*, 38(5):4080–4094, 2022.

[10] M. R Hestenes. Multiplier and gradient methods. *J. Optim. Theory Appl.*, 4(5):303–320, 1969.

[11] D. Fontaine, LaurentMichel, and P. Van Hentenryck. Constraint-based lagrangian relaxation. In *Proc. Int. Conf. on CP: 20th International Conference, CP 2014, Lyon, France, September 8-12, 2014. Proceedings 20*, pages 324–339. Springer, 2014.

[12] C. Coffrin, H. L. Hijazi, and P. Van Hentenryck. The QC relaxation: A theoretical and computational study on optimal power flow. *IEEE Trans. Power Syst.*, 31(4):3008–3018, 2015.

[13] P. Pareek, A. Jayakumar, K. Sundar, D. Deka, and S. Misra. Optimization proxies using limited labeled data and training time–a semi-supervised bayesian neural network approach. *arXiv:2410.03085*, 2024.