TRANSFER LEARNING FOR DATA-DRIVEN POWER FLOW AND OPTIMAL POWER FLOW APPLICATIONS

DORSA NAZARI

A THESIS SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTERS OF SCIENCE

GRADUATE PROGRAM IN ELECTRICAL ENGINEERING & COMPUTER SCIENCE
YORK UNIVERSITY
TORONTO, ONTARIO

FEBRUARY 2025

Abstract

Ensuring real-time grid operations is essential for maintaining both stability and efficiency in today's dynamic power systems. While machine learning (ML)-based approaches enable fast inference, these models are often trained using datasets derived from static grid configurations, such as fixed topologies. Adapting these models to evolving grid conditions introduces additional complexity and necessitates acquiring supplementary datasets, which require computationally intensive solvers.

This paper presents a method that improves computational efficiency compared to conventional transfer learning techniques for adapting ML models used in power flow (PF) and optimal power flow (OPF) analysis to changing grid conditions. Our findings indicate that only 6% of the original dataset requires recalibration, and the entire process of data point regeneration and model fine-tuning is completed in under 3 seconds in the benchmark IEEE 14-bus, IEEE 118-bus, and PEGASE 1352-bus systems. A thorough theoretical, practical, and comparative analysis is provided to demonstrate the effectiveness of the proposed approach.

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Dr. Pirathayini Srikantha, for her invaluable guidance, patience, and support throughout this journey. Her expertise and thoughtful feedback challenged me to refine my work and deepen my understanding. Her encouragement gave me the confidence to overcome obstacles, and I am truly grateful for her mentorship.

I am also profoundly thankful to my family for their unwavering love, sacrifices, and belief in me. Their support has been my foundation, giving me the strength to persevere. This journey has been more than just an academic pursuit; it has been a transformative experience, and I deeply appreciate everyone who has been a part of it.

Table of Contents

List of Tables

List of Figures

List of Symbols

N	Number of buses
$\mathcal{G}$	Generator Bus
$\mathcal{L}$	Load Bus
S_g	Power Generation
S_d	Power Demand
S	Power injection
P	Active power injection
Q	Reactive power injection
V	Voltage
v	Voltage magnitude
θ	Phase angle
I	Current
Y	Admittance matrix
G	Real part of Y matrix
B	Imaginary part of Y matrix
$\mathcal{S}$	Source set
$\mathcal{T}$	Target set
l	Labeled
u	Unlabeled
R	Expected risk

$\mathcal{P}(X, Y)$	New joint distribution
$\mathcal{Q}(X, Y)$	Old joint distribution
$\mathcal{Y}$	Output random variable
$\mathcal{X}$	Input random variable
w	Model parameter
M	Different Topologies
$\mathcal{N}$	Number of training samples
T	Training dataset
h	Hypothesis (model)
J	Objective function
N_{TL}	Required number of data points for transfer learning
SM	Source Model
AM	Adapted Model
TOM	Target-Only Model

Chapter 1

Introduction

We are determined to protect the planet from degradation that it can support the needs of the present and future generations. The shift towards a cleaner and more prosperous economy must be an immediate and ongoing focus in the coming years and decades.

The 2030 agenda for sustainable development [1], endorsed by all United Nations member states in 2015, offers a comprehensive plan for achieving peace and prosperity for both people and the planet, now and into the future. Central to this agenda are the 17 sustainable development goals, which urge all nations, regardless of development status, to take action in a global partnership. Some key goals include:

- Ensuring universal access to affordable, reliable, sustainable, and modern energy.
- Making cities and human settlements inclusive, safe, resilient, and sustainable.

Figure 1.1: Sustainability goals [1]

- Taking urgent actions to combat climate change and its effects.

Projections indicate that earth's temperature will increase by 2035, and vulnerable regions face up to 15 times higher mortality rates due to disasters. To avert the worst impacts of climate change, the government of Canada and 120 countries are committed to achieving net-zero emissions by 2050, meaning their economies will either produce no greenhouse gas emissions or offsets its emissions. [2]

Carbon emissions are calculated by measuring the amount of greenhouse gases (GHGs) released into the atmosphere, expressed in carbon dioxide equivalents (CO_2e) to account for the varying global warming potentials of different gases. In Canada, emissions are tracked using a sector-based inventory consistent with the guidelines set by the Intergovernmental Panel on Climate Change (IPCC). These estimates rely on activity data, such as fuel consumption or industrial production, combined with scientifically established emission factors that quantify the emissions per unit of activity. The data is compiled annually in the National Inventory Report, submitted to the United Nations Framework Convention on Climate Change (UNFCCC) and reviewed for transparency and consistency [3].

Carbon emissions are typically expressed in megatonnes (Mt) of CO_2 equivalent (CO_2e), which accounts for the global warming potential of all greenhouse gases in terms of carbon dioxide. For example, Canada's emissions of 672 Mt CO_2e in 2020 represent the total climate impact of all its greenhouse gas emissions, normalized to CO_2. This metric is used to evaluate environmental impact, monitor progress toward emission targets, and guide climate policy.

Canada's 2030 emissions reduction plan aims to cut greenhouse gas emissions by 40% of levels in 2005 (741 Mt) by 2030. However, as of 2020, emissions were only reduced by 10%, down to 672 Mt, highlighting the need for significant changes across all sectors [2]. The power sector, for instance, must see widespread adoption of renewable energy, electric vehicles, and battery storage systems.

The power grid, as the backbone of the power sector, is a highly complex system, exemplified by the Pan European Grid Advanced Simulation and State Estimation (PEGASE)

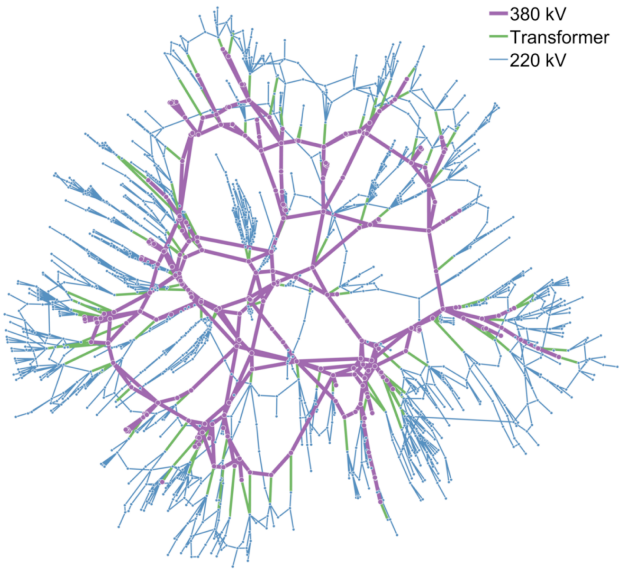


Figure 1.2: 1354 bus system

project [4] in figure 1.2, which includes a 1354-bus portion of the European transmission network. This system accurately reflects the scale and intricacy of the European high-voltage transmission network, consisting of 1,354 buses and approximately 2,000 power lines operating at 380 and 220 kV. The PEGASE 1354 bus system serves as a benchmark frequently utilized by researchers to demonstrate proof of concept due to its complexity and real-world relevance.

Two of the most crucial problems in power grids are Power Flow (PF) and Optimal Power Flow (OPF). Power flow refers to the calculation and analysis of the distribution of electrical power across the various components of a power grid, such as generators, transformers, and transmission lines. It is essential for ensuring that electricity is delivered efficiently and reliably from power plants to consumers. Optimal power flow, on the other hand, takes this a step further by aiming to determine the most efficient operating conditions for the power system. OPF involves minimizing the cost of generation, reducing losses, or achieving other specific objectives while satisfying all operational constraints like voltage limits and power balance. Both PF and OPF are fundamental for maintaining grid stability, optimizing energy use, and ensuring the reliability of electricity supply. However, as networks grow larger, traditional methods for addressing these challenges are increasingly impractical due to their

time-consuming nature.

In response to these challenges, there has been a shift toward innovative approaches that accelerate computational processes. Data-driven methods, particularly those employing machine learning algorithms on historical data, have emerged as a promising alternative. These modern techniques dramatically reduce computation time, making them well-suited for addressing the evolving challenges of complex networks.

Power grid infrastructure undergoes large-scale changes every 10–30 years, such as building new substations or transmission lines. Mid-scale upgrades occur every 5–10 years, while small-scale operational changes, like topology reconfigurations and generation redispatch, happen daily or hourly.

The power grid is continuously evolving, with the addition and removal of power lines for outages, maintenance, and integration of renewable energy sources. Furthermore, uncertainties in demand and supply, as well as line congestions caused by the increasing demand from electric vehicles, add to the challenges.

A significant limitation of data-driven models is their specificity to the grid instances on which they are trained. For example, topological changes in the grid, such as those in the widely researched IEEE 118-bus system with 186 power lines, result in 2^{186} potential configurations. These variations can lead to model drift, where changes in the system topology cause shifts in the underlying data distribution or system behavior, thereby degrading model performance. Generating samples for all possible changes and training a model for each scenario is infeasible. To overcome this limitation, we leverage transfer learning, which allows models to adapt to new but related conditions, thereby enhancing their applicability across different network configurations.

To implement transfer learning, two approaches were utilized: supervised and unsupervised learning. The first approach involved using a small subset of data points from the new topology, while the second approach relied solely on the physical attributes of the power system without utilizing any additional data.

1.1 Literature Review

Power Flow (PF) analysis is crucial for the steady-state operation of electric power systems, underpinning power systems planning, security assessment, and power grid management and optimization. PF analysis is used to solve the steady-state operating conditions of electric power systems and is the most essential tool for power system analysis. PF analysis aims at obtaining complete voltage angles and magnitudes for each bus in a power system, given specified loads, generator real power, and voltage conditions [5].

PF is applied to planning, economic scheduling, system operation, and control areas. Therefore, with the widespread adoption of computer technologies, there has been a constant need to develop efficient power flow analysis methods [6]. The increase in online applications for wide-area monitoring and control has further stressed the necessity of reliable and efficient power flow methodologies. The classic power flow problem (AC model) can be formulated based on Kirchhoff's circuit laws in electric power networks and consists of a set of nonlinear algebraic equations representing real and reactive power injection flow at each node [7].

Optimal Power Flow (OPF) Analysis is an extension of power flow analysis that not only seeks to determine the voltage and power flows but also optimizes certain operational objectives while respecting system constraints. The OPF problem is a fundamental aspect of power system management and market operations, ensuring the high efficiency and security of real-time operations. System operators conduct OPF studies to economically dispatch generation sources, ensuring that power demands are met while stick to system constraints.

This problem presents complex challenges in economic, electrical, and computational aspects. Economically, it necessitates multipart nonlinear pricing to achieve an efficient market equilibrium. Electrically, the use of alternating current (AC) introduces additional nonlinearities. Computationally, the optimization process involves nonconvexities, encompassing both binary variables and continuous functions, complicating the solution process. Moreover, the power system must maintain stability despite the loss of any generator or transmission element [8], [9], [10].

Several algorithms have been developed and refined over the years to solve the AC power flow problem, with the Newton-Raphson (NR) method [11] and the Gauss-Seidel (GS) algorithm [12] being among the most commonly used. NR method is the most powerful algorithm due to its linearizing technique to solve the nonlinear equations using iterative algorithms. However, the inherent characteristics of nonlinear equations in the AC model lead to convergence difficulties and make the process time-consuming, especially when considering contingency analysis [13], [14]. The GS algorithm also iteratively updates the variables associated with each bus type, performing these updates in epochs until convergence is reached. Although the GS algorithm is straightforward and easy to implement, it may exhibit slow convergence, especially in large-scale systems.

To solve the OPF problem, Interior Point Methods (IPM) are a class of optimization techniques. These methods work by navigating through the interior of the feasible solution space, staying clear of the boundary constraints until a solution is reached [15, 16, 17]. IPM operates by iteratively addressing a series of approximations to the original nonlinear problem. Although effective, IPM can be computationally demanding, particularly when applied to very large power systems [18]. The method requires solving substantial linear equation systems at each iteration, which can be both time-intensive and resource-heavy. Furthermore, IPM may encounter challenges with poorly conditioned problems or when the starting point is significantly distant from the optimal solution.

In response, there has been a shift towards innovative approaches that accelerate computation. Data-driven techniques, particularly those employing machine learning (ML) algorithms on historical data, have emerged as a potent alternative [19]. ML provides researchers with a powerful tool to learn the structure of physical phenomenon directly from nature, rather than having to explain the causal relationships through direct application of physics law. Many research and engineering fields, from image recognition to autonomous driving, from health to natural language processing (NLP), have experienced a tremendous boost in performance and efficiency over the last few years. Many problems that seemed impossible to be solved,

can now be tackled thanks to the use of ML methodologies [20].

These ML methods significantly cut down computation time, with some achieving speeds that are orders of magnitude faster than traditional practices, thus offering a promising solution for handling the evolving challenges in PF and OPF analysis for large and complex networks [19].

To solve the PF problem, in [21], a neural network (NN) model is trained to predict PF results using historical power system data. Although the training process may take time, once trained, it is very fast to estimate line flows. A comprehensive performance analysis between the proposed NN-based PF model and the traditional Direct current (DC) power flow model is conducted. The DC model simplifies calculations by assuming uniform voltage magnitudes, ignoring reactive power and losses, resulting in a linear system that can be solved quickly without iterations. While the DC model offers speed and computational efficiency, it lacks the accuracy of the AC model, especially where reactive power and losses are significant. Thus, the AC model is preferred for detailed analysis and planning [13]. It can be concluded that the proposed NN-based PF model can find solutions quickly and more accurately than DC power flow model [21].

Also, research [22] proposes a physics-guided NN to solve the PF problem, with an auxiliary task to rebuild the PF model. By encoding different granularity of Kirchhoff's laws, and system topology into the rebuilt PF model, the NN based PF solver is regularized by the auxiliary task, and constrained by the physical laws.

Besides, to solve the OPF problem, reference [9] shows a novel data-driven approach that combines generative learning, information theory and domain knowledge is proposed to enable real-time OPF studies. Also, [23] presents a survey of recent studies that have applied ML to solve OPF-related problems and provides readers with visions on potential research directions in this field. The surveyed literature is categorized according to the type of studied OPF problems. In [24] the data is used to train a Random Forest to predict solutions of future ACOPF problems. To preserve correlations and relationships between predicted

variables, they utilize a multitarget approach to learn approximate voltage and generation solutions to ACOPF problems directly by only using network loads, without the knowledge of other network parameters or the system topology. they explore the benefits of using the learned solution as an intelligent warm start point for solving the ACOPF.

In addition to supervised methods, there have been several unsupervised approaches. [25] presents an unsupervised deep learning framework designed to generate a subset of decision variables for ACOPF. This framework combines a deep neural network with AC power flow equations, using a fast decoupled power flow solver to assist in calculations. The training process employs a modified augmented Lagrangian function as the loss function, incorporating both generation costs and penalties for constraint violations. To enhance training effectiveness, dynamic adjustment of Lagrangian multipliers is utilized.

The paper by [26] introduces DeepOPF-NGT, an unsupervised learning method designed to address AC optimal power flow challenges. This method focuses on learning the mapping from load conditions to solutions using a specially crafted loss function, tailored for this task, without relying on ground truth data.

[27] proposes an approach that leverages Graph Neural Networks (GNN) as the machine learning model. The training loss combines the ACOPF cost with a weighted penalty for violations of equality and inequality constraints.

However, nowadays due to climate change, the dynamic nature of modern power systems intensified by the incorporation of variable sources like electric vehicles, renewables, and energy storage, has introduced complexities in maintaining system stability. This complexity underscores the necessity for faster PF and OPF computational methods [7]. A significant limitation of current deep neural network (DNN) based approaches is their dependency on a static system topology. The effectiveness of DNN predictions hinges on the assumption that the underlying grid topology remains unchanged. However, as noted in recent studies [28], power grids are undergoing frequent topology reconfigurations to optimize for economic operation and reliability. These changes pose a challenge for traditional DNN models, which

are typically trained for a specific static configuration.

Hence, a significant drawback of these models is their specificity to the grid instances they are trained on. There are intractable number of changes that the model should be able to predict. Several studies have addressed the challenge of adapting to changes in power systems. For example, the method presented in [29] accelerates AC power flow computations by employing machine learning models for initial estimates in the Newton-Raphson process. Although this approach reduces computation times, especially in larger networks, it continues to depend on the Newton-Raphson method, which may be slow under complex conditions. Also, the [30] paper presents a methodology that exhaustively considers all possible topological changes in power networks, requiring a large model and extensive datasets for every conceivable scenario. This brute force approach increases computational demands and complicates practical implementation due to the massive data requirements.

In [31] and [32] typed graph neural network (TGN) model is trained in an unsupervised manner, aiming to minimize the power equilibrium errors across various grid configurations. This approach allows for efficient handling of dynamic changes in the grid topology and can scale effectively with the network size. The performance of the TGN-based solver is validated against traditional methods like the Newton-Raphson solver, demonstrating its capability to provide accurate and reliable results even in complex and changing network conditions. But, this approach is very dependant on Grid Topology Information.

Therefore, to address this limitation, transfer learning (TL) which is adapting models to new, yet related conditions is leveraged. TL enhance the applicability of the model across varying network configurations. It is a powerful machine learning technique that improves the performance of target learners on target domains by leveraging knowledge from related but distinct source domains. By reducing the reliance on extensive labeled data in the target domain, it addresses challenges where collecting sufficient training data is impractical. Transfer learning is versatile, applicable across diverse fields such as natural language processing, computer vision, and sentiment analysis, demonstrating its wide utility. The method not

only enhances learning efficiency but also significantly improves the adaptability of models to new, previously unseen environments [33].

In sentiment analysis, transfer learning helps leverage sentiment knowledge from one domain to improve performance in another, despite differences in language or context. In computer vision, it enables models trained on one set of images to adapt and perform well on different image datasets, enhancing object recognition capabilities. Additionally, in natural language processing, transfer learning aids in tasks like machine translation and speech recognition by using knowledge from one language to assist in processing another, improving accuracy and reducing the need for extensive labeled datasets in the target language [33].

Transfer Learning also has lots of applications in power system. It has significantly enhanced demand response systems by improving electricity demand forecasts and generation predictions from renewable sources. By leveraging models from data-rich environments, it offers better accuracy and efficiency in less mature installations. In reinforcement learning, transfer learning aids in developing advanced control strategies for energy management, using knowledge from related tasks. These applications highlight the method's ability to shorten training times and enhance the predictive capabilities of energy systems [34].

In [35] paper, the authors employ transfer learning to overcome data scarcity challenges in new power systems by leveraging existing data from well-documented grids. This approach enhances event identification capabilities in newer grids by transferring knowledge from a data-rich "source" grid to a data-limited "target" grid. The transfer involves aligning and transforming data through techniques like dimensional reduction and distribution adaptation to apply models effectively across different grids. This method significantly improves the accuracy and efficiency of monitoring and predictive capabilities in power systems with limited historical data.

In [36] paper, transfer learning is applied to enhance Dynamic Security Assessment (DSA) models in power systems, allowing a single model trained on specific faults to assess various unlearned faults efficiently. This method improves the model's ability to generalize across

different scenarios, significantly reducing the need for separate models for each new fault, thus optimizing both computational resources and assessment time.

In [37] paper, Chao Ren and colleagues also employ transfer learning to address the challenges of unlearned faults in power system DSA. The approach allows the adaptation of existing DSA models to new fault types by optimizing the data distribution between known training faults and new, unseen faults. This method enhances the model's ability to accurately predict system security under different fault conditions without the need for extensive retraining. By using transfer learning, the system achieves robust performance even when faced with incomplete data, demonstrating significant improvements over traditional DSA methods.

In the specific field of PF and OPF, the study [38] employs 70 different topologies for pretraining its model, necessitating extensive data that may not be practical in many operational settings. Additionally, it relies on the Newton-Raphson method for data regeneration, which often fails to converge, requiring multiple iterations to achieve a solution.

To improve accuracy, [8] used a pre-trained model as a warm start for re-training the GNN model, incorporating additional data samples generated for the new topology. This re-training step has proven to be extremely efficient in terms of convergence rate. The proposed GNN architecture enables fast transfer learning under topology contingency for the OPF learning problem. For each post-contingency new topology, 4,000 data samples were re-generated, which is half of the original training process, yet still a substantial amount.

It is important to consider that one possible drawback of transfer learning is that the Negative Transfer Learning might happen. Negative transfer occurs when skills or knowledge from one context impair performance in another, often due to differences between the contexts. For example, in machine learning, a model trained on urban images might perform poorly when applied to rural landscapes, demonstrating the challenges of negative transfer [39].

1.2 Problem Statement and Thesis Objective

In this thesis two Transfer Learning based approaches will be considered:

First, we introduce an innovative approach for efficiently adapting machine learning models to altered grid topologies with a small number of data points. It involves taking a pre-trained model (typically trained on a large, general dataset derived from the initial topology) and continuing to train it on a new, often smaller, dataset that is specific to the changed topology. This approach can significantly reduce the time and resources required to develop effective models, especially in cases where the new dataset might be too small to train a model from scratch effectively.

In addition, a technique called reverse sampling is employed in power flow (PF) and optimal power flow (OPF) scenarios. By generating a limited number of new training examples using straightforward one-time computations (matrix multiplications), our approach capitalizes on the intrinsic electrical relationships within the grid. These newly created data points are then used to adjust the original machine learning model. Through detailed theoretical analysis, we demonstrate that our method effectively captures the impact of alterations in grid topology, as the data distributions before and after the topology changes remain closely aligned and negative transfer learning is avoided.

This approach demonstrates that a purely data-driven model can achieve accurate results independently, without relying on traditional iterative methods, potentially leading to significant improvements in computation speed. By training the initial model on a single topology, it requires less data and allows for a simpler architecture, resulting in faster training. This method not only streamlines the training process but also enhances the model's applicability and performance under dynamic conditions. Our approach leverages data from one scenario and dynamically generates data for others, creating a simpler and more manageable model that is more feasible for real-world applications.

Key features of our approach include high-speed adaptation to new grid scenarios, reduced data needs, and minimal training and retraining requirements, all underpinned by the

strengths of transfer learning. This results in a cost-effective, reliable, and accurate power flow computation in dynamic scenarios, which is particularly crucial for grids experiencing branch outages and load growth.

The second approach advances to a stage where no new data is required for adapting to different topologies. To address the challenge of insufficient data for training on every possible anomaly or system change, we are developing models capable of refining their performance through unsupervised learning. We have explored domain adaptation, a technique within unsupervised transfer learning, designed to adapt a model from a source domain to a target domain with different statistical properties. This method is particularly challenging as it does not utilize labeled data in the target domain.

A crucial element of this initiative is the incorporation of the power balance equation directly into the loss function. By embedding fundamental physical constraints, the model ensures adherence to essential power system laws, even in the absence of changed topology data. Additionally, we applied transfer learning techniques, leveraging insights from supervised learning tasks that utilize labeled data to enhance the model's performance in subsequent unsupervised tasks where such labels are absent. The primary objective was to use a supervised learning model as a baseline, facilitating the training of an unsupervised model for adapting to new power system topologies. Initially, the supervised model provided a strong foundation. The unsupervised model, guided by the weights from the supervised model, was then trained to adapt to topology changes. The incorporation of the power balance equation within the loss function played a pivotal role in ensuring the model's accuracy and reliability during this adaptation process.

The main areas of the work plan under the proposed study are as follows for both PF and OPF problems:

i. The dataset, simulated with Matpower, will be preprocessed to ensure it is suitable for analysis.

ii. An initial data-driven model will be trained to establish the mapping between inputs and

outputs. This foundational framework will be enhanced by selecting optimal hyperparameters through grid search.

iii. A novel algorithm is proposed to generate data points for the new topology of the system. This will use a subset of the previous dataset and leverage electrical dependencies to ensure accuracy and relevance.

iv. Theoretically, the similarity of distributions between the original and generated datasets will be demonstrated, affirming the effectiveness of the method in preventing negative transfer learning.

v. The pre-trained model will be fine-tuned using the newly generated dataset to adapt it to the new system topology.

vi. The fine-tuned model will be rigorously tested, and the results will be thoroughly documented and reported.

vii. Comparative analysis will be conducted to evaluate the results against existing benchmarks and analyze the system stability requirements, ensuring the model meets the necessary criteria for real-world application.

viii. Finally, an unsupervised learning approach will be implemented, where the power balance equation is directly incorporated into the loss function, allowing the model to adapt to new system topologies without relying on topology-specific data.

1.3 Thesis Organization

Chapter 2 provides a comprehensive background on power grids, power flow studies, and optimal power flow. It begins by introducing the fundamental components of power grids, including the classification of buses and key variables in power system analysis. The chapter then explores traditional solvers for PF and OPF problems, discussing their methodologies and limitations. Next, it examines data-driven solvers and highlights the challenges associated with applying these methods to new power system. The chapter concludes with an overview

of transfer learning, covering its fundamental concepts, the phenomenon of negative transfer, and the evaluation metrics used to assess model performance.

Chapter 3 introduces transfer learning as a technique for improving power flow problem-solving. It first establishes the methodology, starting with the source model and the process of adapting to new grid topologies. The experimental section follows, discussing the environment and datasets used, evaluating the performance of the models, and analyzing the effectiveness of adaptation strategies. Finally, the chapter compares different methodologies to assess the impact of transfer learning on power flow problem.

Chapter 4 extends the application of transfer learning to optimal power flow problems. The methodology section details the development of the source model and strategies for adapting it to new grid conditions. In the experimental section, the chapter evaluates various datasets, analyzes the transferability of models, and presents a comparative assessment of performance metrics to determine the effectiveness of transfer learning in OPF.

Chapter 5 concludes the thesis by summarizing the key findings and contributions of the research. It reflects on the effectiveness of transfer learning in power flow and optimal power flow problems, highlighting the improvements achieved in model generalization and computational efficiency. The chapter also outlines potential future research directions.

Chapter 2

Background

2.1 Power Grid

A power grid, also known as an electrical grid, is a complex network that delivers electricity from producers to consumers. It comprises three main components: generation, transmission, and distribution systems. The grid is designed to ensure the reliable and efficient delivery of electrical energy, which is essential for modern society's functioning [40].

- **Generation:**

  Electrical power generation is the first stage in the power grid. It involves converting energy from natural resources such as coal, natural gas, nuclear, hydro, wind, and solar into electrical energy. These power plants are typically located far from consumption centers, requiring an extensive transmission network to deliver the generated power [41].

- **Transmission:**

  The transmission system is responsible for transporting electricity over long distances at high voltages, typically between 110 kV and 765 kV. High voltage is used to minimize energy losses due to resistance in the transmission lines. The transmission network

consists of substations, transformers, and high-voltage power lines. Substations play a crucial role in stepping up the voltage for efficient long-distance transmission and stepping it down for distribution to consumers [12].

- **Distribution:**

  Once electricity reaches the vicinity of consumption centers, it enters the distribution system. The voltage is reduced to safer levels, typically between 120 V and 33 kV, depending on the type of consumers (residential, commercial, or industrial). The distribution network then carries the electricity to homes, businesses, and factories. Distribution networks are often radial, with each consumer connected to a single source of supply, although more complex configurations can be used for higher reliability [42].

2.1.1 Types of Buses in Power Systems

In power system analysis, a bus (or node) represents a point where one or more components are connected, such as generators, loads, or transmission lines. Buses are fundamental in solving the power flow equations and understanding the operation of the power grid. There are three primary types of buses, each characterized by specific variables: Load Bus (PQ Bus), Generator Bus (PV Bus), and Slack Bus (Reference Bus).

- **Load Bus (PQ Bus):**

  A Load Bus, also known as a PQ Bus, is where the active power (P) and reactive power (Q) are specified. At this type of bus, the voltage magnitude (V) and phase angle (θ) are unknown and must be calculated. The governing equations at a Load Bus are:

$$P_i = \sum_{j=1}^{n} V_i V_j \left(G_{ij} \cos(\theta_i - \theta_j) + B_{ij} \sin(\theta_i - \theta_j)\right) \tag{2.1}$$

$$Q_i = \sum_{j=1}^{n} V_i V_j \left(G_{ij} \sin(\theta_i - \theta_j) - B_{ij} \cos(\theta_i - \theta_j)\right) \tag{2.2}$$

Where:

- P_i: Active power at bus i
- Q_i: Reactive power at bus i
- V_i: Voltage magnitude at bus i
- θ_i: Voltage phase angle at bus i
- G_{ij}, B_{ij}: Conductance and susceptance between buses i and j

- **Generator Bus (PV Bus):**

  A Generator Bus, or PV Bus, is where the active power (P) and voltage magnitude (V) are specified. At this bus, the reactive power (Q) and voltage phase angle (θ) are unknown and must be determined. The reactive power at a PV Bus is calculated using:

$$Q_i = \sum_{j=1}^{n} V_i V_j \left(G_{ij} \sin(\theta_i - \theta_j) - B_{ij} \cos(\theta_i - \theta_j) \right) \tag{2.3}$$

  The phase angle θ_i is found by solving the power flow equations.

- **Slack Bus (Reference Bus):**

  The Slack Bus, also known as the Reference Bus, serves as a reference point for the system. At this bus, both the voltage magnitude (V) and the phase angle (θ) are specified. The Slack Bus is responsible for balancing the active and reactive power in the system, absorbing any mismatches in power flow calculations. Typically, a generator with a large capacity is selected as the Slack Bus.

2.1.2 Variables in Power System Analysis

- **Admittance Matrix (Y Matrix):**

  In a power system, each transmission line connecting two buses, say bus i and bus k, has an admittance denoted by y_{ik}. This admittance represents how easily current can

flow through the transmission line between these two buses and is the reciprocal of the line's impedance Z_{ik}. The impedance Z_{ik} can be calculated as follows:

$$y_{ik} = \frac{1}{Z_{ik}} = \frac{1}{R_{ik} + jX_{ik}} = G_{ik} + jB_{ik} \tag{2.4}$$

Where:

- R_{ik}: Resistance between buses i and k
- X_{ik}: Reactance between buses i and k
- G_{ik}: Conductance between buses i and k
- B_{ik}: Susceptance between buses i and k

For simplicity in power flow studies, the nodal admittance matrix, also known as the Y-bus matrix, is often used instead of individual impedance or admittance values for each transmission line. The Y-bus matrix not only represents the network's topology but also includes the line admittance and the shunt admittance at each bus. This makes it a powerful tool for analyzing the steady-state operation of the power system.

The nodal admittance matrix (Y-bus matrix) for a grid with N buses is an $N \times N$ matrix where the diagonal elements represent the self-admittance of the bus, and the off-diagonal elements represent the negative of the line admittance between two buses:

$$Y_{ik} = \begin{cases} \sum_{j=1, j\neq i}^{N} y_{ij} + y_{\text{shunt},i}, & \text{if } i = k \\ -y_{ik}, & \text{if } i \neq k \end{cases} \tag{2.5}$$

Where:

- y_{ik}: Line admittance between bus i and bus k
- $y_{\text{shunt},i}$: Shunt admittance at bus i

The Y-bus matrix can be written as:

$$\mathbf{Y} = \begin{bmatrix} Y_{11} & Y_{12} & \cdots & Y_{1N} \\ Y_{21} & Y_{22} & \cdots & Y_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ Y_{N1} & Y_{N2} & \cdots & Y_{NN} \end{bmatrix} \tag{2.6}$$

By assembling the Y-bus matrix, we can simplify power flow analysis and other related calculations, making it easier to study and optimize the operation of the power grid [43].

- **Active Power (P):**

  Active power, denoted by P, is the real component of electrical power that performs useful work, such as turning motors or lighting lamps. It is measured in watts (W) and is calculated using the following equation:

$$P_i = V_i \sum_{j=1}^{n} V_j \left(G_{ij} \cos(\theta_i - \theta_j) + B_{ij} \sin(\theta_i - \theta_j) \right) \tag{2.7}$$

  Where:

  - P_i: Active power at bus i
  - V_i: Voltage magnitude at bus i
  - θ_i: Voltage phase angle at bus i
  - G_{ij}, B_{ij}: Conductance and susceptance between buses i and j

- **Reactive Power (Q):**

  Reactive power, denoted by Q, is the imaginary component of electrical power that does not perform useful work but is necessary for maintaining the voltage levels in the system. It is measured in volt-amperes reactive (VAR) and is essential for the operation

of inductive loads like motors and transformers. The reactive power at a bus can be calculated using:

$$Q_i = V_i \sum_{j=1}^{n} V_j \left(G_{ij} \sin(\theta_i - \theta_j) - B_{ij} \cos(\theta_i - \theta_j) \right) \tag{2.8}$$

- **Voltage Magnitude (V) and Phase Angle (θ):**

  Voltage magnitude, denoted by V, represents the absolute value of the voltage at a bus. It is a crucial variable in ensuring that electrical power is delivered efficiently and safely across the grid. The phase angle, denoted by θ, represents the angle of the voltage phasor relative to a reference, usually the Slack Bus. The difference in phase angles between buses drives the flow of active and reactive power across the network [44].

2.1.3 Example

The IEEE 14-bus system [45] is a widely used benchmark in power system studies. It consists of 14 buses, 5 generators, 11 loads, and 20 transmission lines. The system includes three types of buses: Load (PQ) Buses, Generator (PV) Buses, and a Slack Bus, as described below.

Types of Buses in IEEE 14-Bus System

- **Slack Bus:**

  Bus 1 in the IEEE 14-bus system is the Slack Bus, which serves as the reference bus. The voltage magnitude and phase angle are specified at this bus, and it absorbs any active and reactive power imbalances in the system.

  - Voltage magnitude (V): 1.06 p.u.
  - Phase angle (θ): 0 degrees

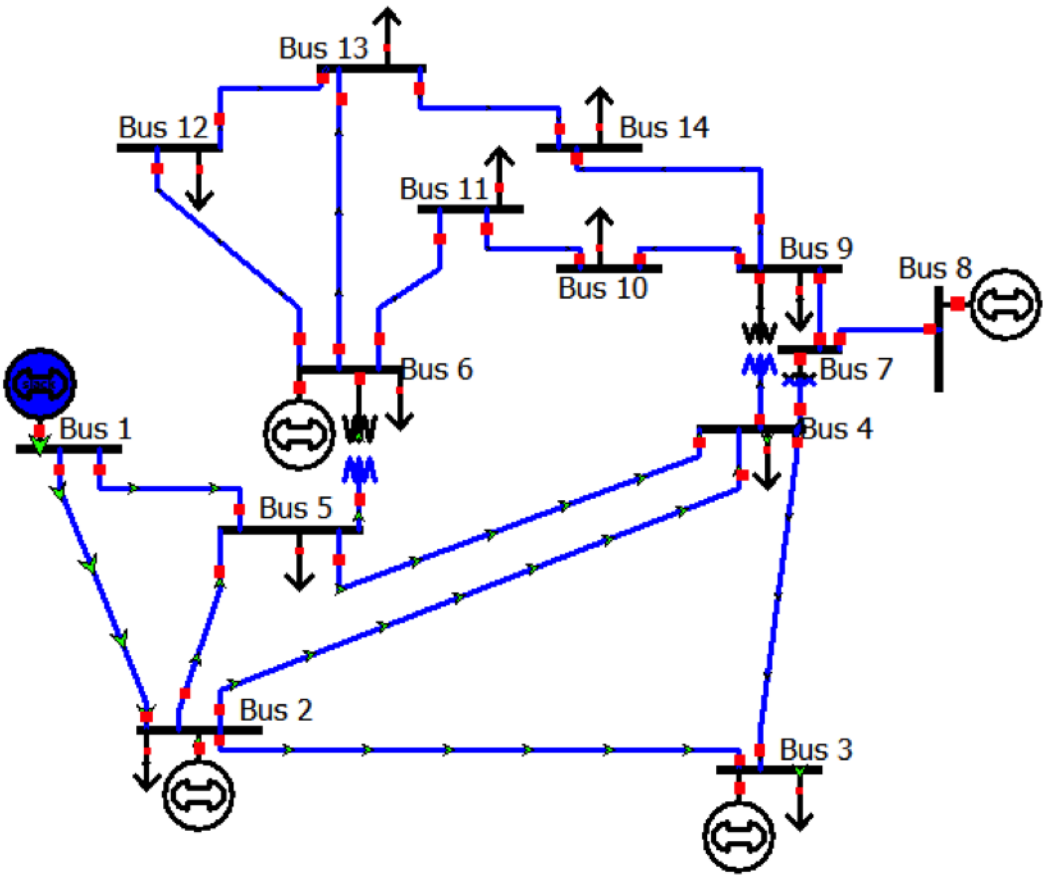


Figure 2.1: IEEE 14 case

- **Generator Buses (PV Buses):**

  The IEEE 14-bus system has 4 PV Buses (Buses 2, 3, 6, and 8), where the active power (P) and voltage magnitude (V) are specified, and the reactive power (Q) and phase angle (θ) are calculated.

  For example, at Bus 2:

  - Specified active power (P): 40 MW
  - Voltage magnitude (V): 1.045 p.u.
  - Reactive power (Q) and phase angle (θ) are calculated through power flow analysis.

- **Load Buses (PQ Buses):**

  The remaining buses in the IEEE 14-bus system are Load Buses, where the active power (P) and reactive power (Q) are specified, while the voltage magnitude (V) and phase angle (θ) need to be calculated. For example, at Bus 4:

  - Specified active power (P): 70 MW
  - Specified reactive power (Q): 25 MVAR
  - Voltage magnitude (V) and phase angle (θ) are calculated through power flow analysis.

2.2 Power Flow Studies

Power flow analysis is a key task in electrical power system studies. It focuses on determining the voltages, currents, and power flows throughout an electrical grid for a specified load and generation condition. As mentioned, each bus has four primary variables: active power, reactive power, voltage magnitude, and voltage angle. The main objective of a power flow study is to calculate these variables at each bus in the network based on power flow equations.

This study forms the foundation for optimal power flow solutions, as any feasible solution must satisfy the power flow equations at every bus.

The general power flow equations are given by:

$$P_i = V_i \sum_{j=1}^{n} V_j \left(G_{ij} \cos(\theta_i - \theta_j) + B_{ij} \sin(\theta_i - \theta_j) \right) \tag{2.9}$$

$$Q_i = V_i \sum_{j=1}^{n} V_j \left(G_{ij} \sin(\theta_i - \theta_j) - B_{ij} \cos(\theta_i - \theta_j) \right) \tag{2.10}$$

Where:

- P_i: Active power at bus i
- Q_i: Reactive power at bus i
- V_i: Voltage magnitude at bus i
- θ_i: Voltage phase angle at bus i
- G_{ij}, B_{ij}: Elements of the Y matrix, representing conductance and susceptance between buses i and j
- n: Total number of buses in the system

2.2.1 Bus Injection Model

The power flowing between buses i and k is represented as the complex power S_{ik}, which can be separated into its real (active) power P_{ik} and reactive power Q_{ik} components:

$$S_{ik} = P_{ik} + jQ_{ik} \tag{2.11}$$

Alternatively, it can be expressed as the product of the voltage at bus i (V_i) and the conjugate of the current flowing from bus i to k (I_{ik}^*):

$$S_{ik} = V_i I_{ik}^* \tag{2.12}$$

According to Ohm's Law, the current between two buses is proportional to the potential difference and inversely proportional to the impedance between them. Since admittance is the inverse of impedance, the current is:

$$I_{ik} = y_{ik}(V_i - V_k) \tag{2.13}$$

Using the Y-bus matrix, the relationship between current and voltage at all buses can be expressed as:

$$\mathbf{I} = \mathbf{Y}_{bus}\mathbf{V} \tag{2.14}$$

$$\begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_N \end{bmatrix} = \begin{bmatrix} Y_{11} & Y_{12} & \cdots & Y_{1N} \\ Y_{21} & Y_{22} & \cdots & Y_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ Y_{N1} & Y_{N2} & \cdots & Y_{NN} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_N \end{bmatrix}$$

Where I_i is the current at bus i, and V_i is the voltage at the same bus. For any bus i, the injected current can be expressed as:

$$I_i = \sum_{k=1, k\neq i}^{N} y_{ik}(V_i - V_k) \tag{2.15}$$

Kirchhoff's Current Law (KCL) states that the sum of currents entering and leaving a node must be equal. This principle underpins the bus injection model, which states that the total power injected into any bus i must equal the total power leaving that bus. Mathematically, this can be written as:

$$S_{Gi} - S_{Di} = V_i \sum_{k=1}^{N} Y_{ik}^* V_k^* \tag{2.16}$$

Where:

- S_{Gi} is the power generated at bus i
- S_{Di} is the power consumed at bus i
- Y_{ik}^* is the conjugate of the admittance between buses i and k

This equation can be split into active and reactive power components as follows:

$$P_{Gi} - P_{Di} = |V_i| \sum_{k=1}^{N} |V_k| \left(G_{ik} \cos(\phi_i - \phi_k) + B_{ik} \sin(\phi_i - \phi_k)\right) \tag{2.17}$$

$$Q_{Gi} - Q_{Di} = |V_i| \sum_{k=1}^{N} |V_k| \left(G_{ik} \sin(\phi_i - \phi_k) - B_{ik} \cos(\phi_i - \phi_k)\right) \tag{2.18}$$

Where:

- P_{Gi} and Q_{Gi} are the active and reactive power generation at bus i
- P_{Di} and Q_{Di} are the active and reactive power demands at bus i
- G_{ik} and B_{ik} represent the conductance and susceptance between buses i and k
- ϕ_i and ϕ_k are the voltage phase angles at buses i and k

These equations form the basis for power flow analysis and help in determining the steady-state behavior of electrical networks.

2.3 Optimal Power Flow Studies

2.3.1 Formulation of OPF

The Optimal Power Flow (OPF) problem was initially introduced in the 1960s [46], and it has since been a fundamental tool for system operators, such as Independent System Operators (ISOs), to determine the most cost-effective dispatch of generation units to meet electricity demand. With the rising complexity of modern power grids, including increased demand and the integration of renewable energy sources, the OPF problem has evolved into a critical challenge in economic dispatch for smart grids.

OPF is defined as a constrained optimization problem aiming to minimize a specific objective, such as the total generation cost, while satisfying various system constraints. These constraints include power balance equations (such as those in Equations (2.20) and (2.22)), as well as operational limits on generation capacities, voltage magnitudes, and transmission line flows. Solutions to OPF problems must satisfy two main conditions: (1) *feasibility*, meaning that the solution adheres to all system constraints, and (2) *optimality*, ensuring the solution minimizes the specified cost function.

Standard OPF Problem

The OPF problem can be expressed mathematically as follows:

$$\min \sum_{g \in G} C_g(P_{Gg}) \tag{2.19}$$

where $C_g(P_{Gg})$ represents the cost of real power generated by generator g and is typically modeled as a quadratic function [47].

The problem is subject to the following constraints:

- **Real and reactive power limits**:

$$P_{Gg}^{\min} \leq P_{Gg} \leq P_{Gg}^{\max} \quad \forall g \in G \tag{2.20}$$

$$Q_{Gg}^{\min} \leq Q_{Gg} \leq Q_{Gg}^{\max} \quad \forall g \in G \tag{2.21}$$

- **Power balance equation**: The power injected by the generators must match the demand at each bus $i \in N$:

$$P_{Gi} - P_{Di} + j(Q_{Gi} - Q_{Di}) = V_i \sum_{j \in N} V_j^* Y_{ij}^* \quad \forall i \in N \tag{2.22}$$

- **Voltage magnitude limits**:

$$V_i^{\min} \leq |V_i| \leq V_i^{\max} \quad \forall i \in N \tag{2.23}$$

- **Phase angle limits**:

$$\phi_i^{\min} \leq \phi_i \leq \phi_i^{\max} \quad \forall i \in N \tag{2.24}$$

- **Branch flow limits**: The flow through each transmission line must not exceed thermal limits:

$$|V_i(V_i^* - V_j^*)Y_{ij}^*| \leq S_{ij}^{\max} \quad \forall (i,j) \in N, i \neq j \tag{2.25}$$

In this formulation, the sets G and N represent the generators and buses in the network, respectively. The decision variables include the real and reactive power generated at each bus, as well as the bus voltages and phase angles.

2.3.2 Challenges in Solving OPF

The OPF problem has been shown to be NP-hard in its most general form due to the non-convex nature of several constraints, particularly those related to power balance, voltage magnitudes, and branch flows [48]. The integration of distributed energy resources (DERs) such as wind and solar energy further complicates the problem by introducing uncertainty and variability. Traditionally, OPF is solved over fixed time intervals (e.g., day-ahead, hourly,

or every 5-15 minutes), with the assumption that demand changes gradually over these periods. However, this assumption is becoming less valid with the growing use of DERs and fluctuating consumption patterns, making real-time OPF solutions an important area of ongoing research [49].

2.4 Traditional Solvers for PF and OPF Problems

Power Flow (PF) and Optimal Power Flow (OPF) problems are fundamental to the analysis and operation of power systems. Power Flow (PF) problems deal with determining the voltage magnitude and phase angle at each bus in the system, while Optimal Power Flow (OPF) problems go a step further by optimizing an objective (such as minimizing generation costs) while satisfying system constraints. Over the years, various traditional solvers have been developed to address these problems efficiently.

2.4.1 Traditional Solvers for Power Flow (PF)

The Power Flow problem is typically solved using the following traditional methods:

Newton-Raphson Method

The Newton-Raphson method is the most widely used iterative technique for solving the non-linear power flow equations due to its robustness and fast convergence for well-conditioned systems. It linearizes the power flow equations through a first-order Taylor expansion and iteratively solves the system. This method provides fast convergence and high accuracy, especially for systems where the initial guess is close to the actual solution [47].

Gauss-Seidel Method

The Gauss-Seidel method is an older iterative method for solving power flow problems. It updates each bus's voltage based on the previous iteration's values, using a direct substitution

approach. Although it is simple to implement and requires minimal computational resources, it converges more slowly than Newton-Raphson, particularly for large or complex systems [47].

Fast Decoupled Power Flow (FDPF)

The Fast Decoupled Power Flow (FDPF) is an approximation of the Newton-Raphson method that decouples the real and reactive power calculations. This leads to faster computation times, making it more suitable for large systems. However, it is less accurate in systems where the assumptions of small angle differences do not hold [47].

2.4.2 Traditional Solvers for Optimal Power Flow (OPF)

The Optimal Power Flow (OPF) problem incorporates both the nonlinear power flow equations and an objective function, such as minimizing the cost of power generation, subject to operational constraints.

Gradient-Based Methods

Gradient-based methods use the gradient of the objective function to guide the optimization process. Newton's method is often extended to handle the optimization problem with constraints. While efficient when gradients can be computed, they may face difficulties in handling non-convex problems and may converge to local optima [46].

Interior Point Methods (IPM)

Interior Point Methods (IPMs) are widely used for OPF problems. These methods solve large-scale nonlinear optimization problems by iterating within the interior of the feasible region. IPMs are known for efficiently handling complex, large-scale systems and constraints, and they are especially suitable for both convex and non-convex problems [48].

Sequential Quadratic Programming (SQP)

Sequential Quadratic Programming (SQP) solves the OPF problem by iterating over a sequence of quadratic subproblems that approximate the original nonlinear problem. SQP is known for being highly accurate but computationally intensive, especially for large systems [47].

Linear Programming (LP) and Quadratic Programming (QP)

In some cases, the OPF problem can be linearized (e.g., in DC OPF) or approximated as a quadratic programming problem, enabling the use of LP and QP solvers. These methods offer computational efficiency but are less accurate when approximations cannot fully capture the nonlinearity of the system [50].

Lagrangian Relaxation

Lagrangian Relaxation decomposes the OPF problem into smaller subproblems by relaxing the coupling constraints. This method is especially useful for large, decentralized systems. However, it may provide suboptimal solutions if the relaxed problem does not converge [47].

2.4.3 Comparison of Traditional Solvers

Table 2.1 provides a comparison of the traditional solvers for PF and OPF problems, highlighting their strengths and weaknesses.

2.5 Data Driven Solvers

The growing complexity of modern power systems, which includes increasing integration of renewable energy sources, distributed energy resources (DERs), and rapidly fluctuating demand patterns, has necessitated the development of more efficient and adaptive approaches for solving Power Flow (PF) and Optimal Power Flow (OPF) problems. Traditional solvers for

Table 2.1: Comparison of Traditional Solvers for PF and OPF Problems

Solver	**Problem Type**	**Main Features**	**Strengths/Weaknesses**
Newton-Raphson	PF	Iterative, uses Jacobian matrix	Fast convergence, high accuracy / Sensitive to poor initial guesses [47]
Gauss-Seidel	PF	Iterative, direct updating	Simple, low memory usage / Slow convergence [47]
Fast Decoupled Power Flow (FDPF)	PF	Approximated power flow solution	Computationally efficient / Less accurate in large angle difference systems [47]
Gradient-based Methods	OPF	Optimization using gradient and Hessian	Efficient for smooth problems / May converge to local optima [46]
Interior Point Methods (IPM)	OPF	Solves large-scale nonlinear optimization problems	Efficient for large-scale systems / Computational complexity for small systems [48]
Sequential Quadratic Programming	OPF	Solves a series of quadratic subproblems	Accurate, reliable / Computationally intensive for large systems [47]
Linear Programming (LP) / Quadratic	OPF	Solves linear or quadratic approximations	Fast, efficient / Limited to linear or quadratic approximations [50]
Lagrangian Relaxation	OPF	Decomposes problem into smaller subproblems	Good for decentralized systems / May provide suboptimal solutions [47]

these problems, such as Interior Point Methods (IPM) and Sequential Quadratic Programming (SQP), struggle to meet the demands of real-time operation and to handle the high levels of uncertainty inherent in renewable energy integration. In this context, **data-driven solvers** have emerged as an effective alternative to traditional methods. These solvers rely on historical data and machine learning techniques to improve computational efficiency and adaptability, making them suitable for modern smart grids [48, 51].

2.5.1 Key Concepts of Data-Driven Solvers

Data-driven solvers for PF and OPF rely on large amounts of operational data from smart meters, sensors, and phasor measurement units (PMUs). This data is used to train machine learning (ML) models, enabling them to approximate complex non-linear relationships between input features (such as demand, generation, and voltage) and output variables (such as optimal power dispatch). These solvers have gained popularity due to their ability to

provide fast, accurate, and scalable solutions, particularly for real-time applications.

2.5.2 Machine Learning Models in Power Systems

Several machine learning approaches have been applied to solve PF and OPF problems:

- **Supervised Learning**: Supervised learning techniques are used to learn mappings from input features (such as power demand and renewable generation) to output variables (such as voltage). These models are trained on historical PF and OPF data and provide fast approximations of solutions when applied in real-time.

- **Unsupervised Learning**: Clustering and dimensionality reduction techniques are employed to uncover patterns in the data, which helps simplify the complexity of the optimization problem. By reducing the dimensionality of the feature space, the computational effort required to solve the OPF problem is reduced.

- **Reinforcement Learning (RL)**: RL is particularly suited for dynamic and sequential decision-making problems, such as energy dispatch or voltage control. In this approach, an agent learns optimal strategies by interacting with the power grid environment and receiving feedback (rewards) based on actions taken, such as minimizing generation costs or maintaining voltage stability.

2.5.3 Deep Learning for Nonlinear Power Systems

Deep learning techniques, particularly **deep neural networks (DNNs)**, are used to approximate highly nonlinear relationships in PF and OPF problems. These models are trained on large datasets of historical system states and provide rapid predictions of optimal solutions. This is particularly useful for real-time applications, where traditional methods might be computationally prohibitive.

Additionally, hybrid approaches that combine DNNs with traditional solvers (such as Newton-Raphson or Sequential Quadratic Programming) offer a promising solution. In these

methods, DNNs provide a fast, approximate solution, which can be refined by a traditional solver to ensure accuracy and adherence to constraints.

2.5.4 Applications of Data-Driven Solvers in Power Systems

Data-driven solvers offer significant advantages in several applications within power systems:

- **Real-Time OPF**: Data-driven solvers can predict near-optimal solutions for OPF problems in real time, making them suitable for time-sensitive grid operations. Traditional solvers, such as IPMs, may take several minutes to converge, but data-driven approaches, once trained, can provide solutions in milliseconds.
- **Handling Uncertainty**: By learning from historical data, data-driven models can account for uncertainties in renewable generation and demand, thus offering more robust solutions than traditional solvers, which often rely on deterministic models.
- **Predictive Control**: Data-driven solvers, especially those that employ RL, enable predictive control, where actions are taken based on forecasts of future system states. This is essential for grid operations where proactive measures are needed to maintain stability under fluctuating demand and renewable generation conditions.

2.5.5 Challenges and Future Directions

Despite their advantages, data-driven solvers also face several challenges:

- **Data Quality and Availability**: The effectiveness of these solvers depends heavily on the quality and quantity of available data. Incomplete or noisy data can significantly reduce model accuracy.
- **Generalization**: ML models may struggle to generalize well to unseen conditions, such as extreme weather events or blackouts, which may not be present in the training data [51].

- **Interpretability**: Many data-driven models, particularly deep learning models, function as "black boxes," making their decision-making process difficult to interpret. This lack of transparency can hinder their adoption in critical power system operations.

2.6 Transfer Learning

Transfer learning is a machine learning technique where a model developed for a particular task is reused as the starting point for a model on a second task. This approach leverages the knowledge gained from one task (source task) to improve the performance or reduce the learning time on a new task (target task). Transfer learning is particularly useful when the target task has limited labeled data, and the source task has a large amount of data [52].

2.6.1 Notation

Based on [53, 54], let $\mathcal{X}$ be the input feature space and $\mathcal{Y}$ the output label space ($\mathcal{Y} \subseteq \mathbb{R}$ for regression). We will use $P_{\mathcal{S}}(X, Y)$ and $P_{\mathcal{T}}(X, Y)$, respectively, to denote the joint distribution in the source and the target domain, where X is the input random variable and Y the output. Following the convention, we assume having access to a labeled source set $\mathcal{S} = \{(x_s^i, y_s^i)\}_{i=1}^{n_s}$ sampled from the source joint $P_{\mathcal{S}}(X, Y)$, a labeled target set $\mathcal{T}_l = \{(x_l^j, y_l^j)\}_{j=1}^{n_l}$ drawn from the target joint $P_{\mathcal{T}}(X, Y)$, and an unlabeled target set $\mathcal{T}_u = \{x_u^k\}_{k=1}^{n_u}$ from the target marginal $P_{\mathcal{T}}(X)$. For convenience, we define $\mathcal{T} = (\mathcal{T}_l, \mathcal{T}_u)$. In unsupervised TL, the target set only has unlabeled samples, i.e., $\mathcal{T} = \mathcal{T}_u$.

Under this notation, transfer learning aims at designing an algorithm A, which takes both the source and target domain data $\mathcal{S}$, $\mathcal{T}$ as input, and outputs a better hypothesis (model) $h = A(\mathcal{S}, \mathcal{T})$, compared to only using the target-domain data. For model comparison, we will adapt the standard expected risk, which is defined as

$$R(h) := \mathbb{E}_{x,y \sim P_T} \left[l(h(x), y) \right], \tag{2.26}$$

with l being the specific task loss. To make the setting meaningful, it is often assumed that $n_s \gg n_l$.

2.6.2 Negative Transfer (NT)

While transfer learning can significantly enhance model performance by utilizing knowledge from related domains, it also carries the risk of negative transfer, where inappropriate knowledge application can degrade performance. Negative transfer typically occurs when the source and target domains are not sufficiently related, leading to misleading or inaccurate predictions. This phenomenon can be detrimental, as it may not only fail to improve model performance but can actively make it worse. To prevent negative transfer, it is crucial to carefully assess the relevance and compatibility between the source and target domains [33].

A widely accepted description of negative transfer is: "transferring knowledge from the source can have a negative impact on the target learner." While intuitive, this description conceals many critical factors underlying negative transfer, among which we stress the following three points:

1. **Algorithm-Specific Definition:** To study negative transfer, one should focus on a specific algorithm and compare its performance with and without the source-domain data. Hence, we define the negative transfer condition (NTC) for any algorithm A as:

$$R_{P_T}(A(\mathcal{S}, \mathcal{T})) > R_{P_\mathcal{T}}(A(\theta, \mathcal{T})). \tag{2.27}$$

   For convenience, we also define the negative transfer gap (NTG) as a quantifiable measure of negative transfer:

$$\text{NTG} = R_{P_\mathcal{T}}(A(\mathcal{S}, \mathcal{T})) - R_{P_\mathcal{T}}(A(\theta, \mathcal{T})), \tag{2.28}$$

   and we say that negative transfer occurs if the NTG is positive and vice versa. In

this notation, $R_{P_T}(\cdot)$ denotes the expected risk (or error) over the target distribution $P_{\mathcal{T}}$, and $A(\mathcal{S}, \mathcal{T})$ represents the hypothesis learned by algorithm A using both source dataset $\mathcal{S}$ and target dataset $\mathcal{T}$. The term $A(\theta, \mathcal{T})$ denotes the hypothesis learned using only the target dataset $\mathcal{T}$, where θ represents a randomly initialized model.

2. **Divergence Between Joint Distributions:** The divergence between the joint distributions $P_{\mathcal{S}}(X, Y)$ and $P_{\mathcal{T}}(X, Y)$ is the root cause of negative transfer. As negative transfer is algorithm-specific, it is natural to ask whether there exists a transfer learning algorithm that can always improve the expected risk compared to its target-domain-only baseline. This depends on the divergence between $P_{\mathcal{S}}(X, Y)$ and $P_{\mathcal{T}}(X, Y)$. For example, if $P_{\mathcal{S}}(X) = P_{\mathcal{T}}(X)$ but $P_{\mathcal{S}}(Y|X)$ is uniform for any X, there is no meaningful knowledge in $P_{\mathcal{S}}(X, Y)$, and exploiting $\mathcal{S} \sim P_{\mathcal{S}}(X, Y)$ will almost surely harm the estimation of $P_{\mathcal{T}}(Y|X)$ unless $P_{\mathcal{T}}(Y|X)$ is also uniform.

   In practice, if there is some systematic similarity between $P_{\mathcal{S}}(X, Y)$ and $P_{\mathcal{T}}(X, Y)$, an ideal transfer would take advantage of the similar part, leading to improved performance. However, if an algorithm fails to discard the divergent part, one can expect negative transfer to occur. Thus, regardless of the algorithm choice, the distribution shift is the root cause of negative transfer.

3. **Size of the Labeled Target Data:** Negative transfer largely depends on the size of the labeled target data. An overlooked factor of negative transfer is the amount of labeled target data, which has a mixed effect. For the same algorithm and distribution divergence, the negative transfer condition (NTC) depends on how well the algorithm performs using target data alone, i.e., the right-hand side of Eq. 2.27. In zero-shot transfer learning, where there is no labeled target data ($n_l = 0$), only using unlabeled target data would result in a weak random model, and thus NTC is unlikely to be satisfied. When labeled target data is available, a better target-only baseline can be obtained using semi-supervised learning methods, and negative transfer is relatively

more likely to occur. At the other end of the spectrum, if there is an abundance of labeled target data, transferring from a slightly different source domain could hurt generalization.

The amount of labeled target data also affects the feasibility and reliability of discovering shared regularity between the joint distributions $P_{\mathcal{S}}(X, Y)$ and $P_{\mathcal{T}}(X, Y)$. When labeled target data is unavailable ($n_l = 0$), one must rely on the similarity between the marginals $P_{\mathcal{S}}(X)$ and $P_{\mathcal{T}}(X)$, which has theoretical limitations. In contrast, with a sufficient number of samples from $P_{\mathcal{T}}(X, Y)$ and $P_{\mathcal{S}}(X, Y)$, the problem becomes more manageable, and an ideal transfer learning algorithm may utilize labeled target data to mitigate the negative impact of unrelated source information [53].

2.6.3 Methods of Knowledge Transfer

Freezing Layers

Freezing layers is a method where certain layers of a pre-trained model are kept unchanged during the training of the target task. Typically, the early layers of a neural network are frozen because they capture general features that are useful across various tasks. This approach allows the model to retain the low-level representations learned from the source task while focusing the training on the higher-level layers that are specific to the target task [55].

Fine-Tuning

Fine-tuning involves starting with a pre-trained model and then continuing the training process on the target task. Unlike freezing layers, fine-tuning allows all or most of the model's parameters to be updated, albeit often with a lower learning rate. This approach is particularly effective when the source and target tasks are closely related, as it allows the model to adapt to the nuances of the target task while retaining the knowledge acquired from the source task [56, 57].

Feature Extraction

In feature extraction, the pre-trained model is used as a fixed feature extractor. The idea is to leverage the rich set of features learned by the model on the source task to generate a new feature representation for the target task. The extracted features are then fed into a separate classifier (e.g., an SVM or a shallow neural network) that is trained specifically for the target task. This method is particularly useful when the target task has very limited data [58].

2.7 Kullback Leibler Divergence

Kullback-Leibler (KL) divergence, introduced by Kullback and Leibler in 1951 [59], is a fundamental concept in information theory and statistics. It measures the "distance" between two probability distributions and quantifies the amount of information lost when approximating one distribution with another. Formally, KL divergence is not a true metric since it is not symmetric, but it has become a critical tool in areas like machine learning, information theory, and statistical inference.

2.7.1 Definition of KL Divergence

KL divergence measures the difference between two probability distributions P and Q. For discrete probability distributions, it is defined as:

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_{x \in X} P(x) \log \left(\frac{P(x)}{Q(x)} \right)$$

where P and Q are two probability distributions. For continuous distributions, the summation is replaced with an integral:

$$D_{\mathrm{KL}}(P \parallel Q) = \int_{-\infty}^{\infty} p(x) \log \left(\frac{p(x)}{q(x)} \right) dx$$

KL divergence quantifies the difference between the distributions in terms of how much

information is lost when assuming the distribution Q instead of P. A key property of KL divergence is that it is non-negative, i.e., $D_{\mathrm{KL}}(P \parallel Q) \geq 0$, with equality only if $P = Q$ almost everywhere.

2.7.2 Applications of KL Divergence

KL divergence is widely used in various fields. In machine learning, it is often used as a loss function in generative models, such as variational autoencoders, and is used in information theory to measure the expected amount of information gained from a new distribution relative to a prior. In hypothesis testing, KL divergence provides a measure of how different two hypotheses are, and in statistics, it is used to compare probability distributions for tasks like distribution fitting and model selection [59].

2.7.3 Properties of KL Divergence

Kullback and Leibler emphasized the asymmetry of KL divergence, which means that $D_{\mathrm{KL}}(P \parallel Q) \neq D_{\mathrm{KL}}(Q \parallel P)$. This asymmetry highlights that KL divergence is directional—it measures the divergence of one distribution from another in a specific order. Additionally, KL divergence is additive for independent events and remains invariant under sufficient statistics, preserving the information content as long as sufficient statistics are employed [59].

2.8 Evaluation Metrics

In the evaluation of the power flow prediction model, several key metrics are used to assess the performance of the model. These metrics provide insights into the accuracy of the model, its error distribution, and its ability to generalize to unseen data. The primary metrics used in this work are Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), and the R^2 score. Additionally, case-specific metrics like the MSE for voltage magnitudes and voltage angles are employed to further refine the

evaluation. Each metric is explained in detail below.

2.8.1 Mean Squared Error (MSE):

The Mean Squared Error (MSE) is a standard measure of model performance in regression tasks. It calculates the average of the squared differences between the predicted values and the true values. Mathematically, it is defined as:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

where N is the number of data points, y_i is the true value, and $\hat{y}_i$ is the predicted value. MSE penalizes larger errors more than smaller ones due to the squaring of the residuals. A lower MSE indicates a model that makes predictions closer to the actual values. In this work, MSE is used as the primary loss function during training and also as an evaluation metric for both voltage magnitudes and angles.

2.8.2 Root Mean Squared Error (RMSE):

The Root Mean Squared Error (RMSE) is another common metric in regression analysis, representing the square root of the MSE. It provides an error measure in the same units as the predicted variable, making it easier to interpret in practical terms. The RMSE is defined as:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$$

RMSE is particularly useful for understanding the typical size of the prediction error. Like MSE, a lower RMSE indicates better model performance. RMSE is used in this study to evaluate the overall accuracy of the power flow predictions.

2.8.3 Mean Absolute Percentage Error (MAPE):

The Mean Absolute Percentage Error (MAPE) is a metric that expresses the prediction error as a percentage. It is defined as:

$$\text{MAPE} = \frac{100}{N} \sum_{i=1}^{N} \left| \frac{y_i - \hat{y}_i}{y_i} \right|$$

where y_i is the true value and $\hat{y}_i$ is the predicted value. MAPE provides a measure of error relative to the size of the true values, which can be useful when dealing with values that vary greatly in magnitude. However, it should be noted that MAPE can become skewed if the true values y_i are close to zero. In this work, MAPE is used to assess the model's accuracy in terms of percentage error, providing an intuitive understanding of the model's predictive performance.

2.8.4 R^2 Score:

The R^2 score, also known as the coefficient of determination, measures the proportion of the variance in the dependent variable that is predictable from the independent variables. It is defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{N} (y_i - \bar{y})^2}$$

where $\bar{y}$ is the mean of the true values. The R^2 score ranges from 0 to 1, with a score of 1 indicating perfect prediction and a score of 0 meaning that the model explains none of the variability of the response data. An R^2 score closer to 1 indicates a model with a high level of accuracy. In this study, the R^2 score is used to assess how well the model fits the data.

2.8.5 95% Confidence Interval

The **95% confidence interval (CI)** is a range of values within which the true value of a parameter is expected to lie with 95% confidence. In other words, it gives an estimate of

the uncertainty surrounding a prediction or a measured value. In the context of this work, the 95% confidence interval provides a range in which the true validation loss lies with 95% certainty. A narrow confidence interval indicates high precision in the model's predictions, while a wider interval indicates greater uncertainty.

The confidence interval is computed as follows:

$$\text{CI} = \left(\bar{x} - z_{\alpha/2} \cdot \frac{\sigma}{\sqrt{n}}, \bar{x} + z_{\alpha/2} \cdot \frac{\sigma}{\sqrt{n}} \right)$$

where:

- $\bar{x}$ is the sample mean,
- $z_{\alpha/2}$ is the critical value from the standard normal distribution (for a 95% confidence level, $z_{\alpha/2} = 1.96$),
- σ is the sample standard deviation,
- n is the sample size.

In this work, the 95% confidence interval was used to assess the uncertainty in the model's validation loss. It allows us to determine how reliable the model's predictions are when generalized to unseen data.

2.8.6 Eta Squared (η^2)

The **Eta Squared (η^2)** is a measure of effect size, commonly used in statistical analysis to quantify the proportion of variance in the dependent variable that is explained by the independent variables or the model. In the context of this work, η^2 measures how much of the variability in the predicted values can be attributed to the model's predictions.

The formula for η^2, as used in this study, is as follows:

$$\eta_2 = 1 - \frac{1}{K^{(m)}} \sum_{k=1}^{K^{(m)}} \frac{1}{2|G| - 1} \sum_{d=1}^{2|G|-1} \left| \frac{\hat{y}_{k,d}^{(m)} - y_{k,d}^{(m)}}{y_{k,d}^{(m)}} \right|$$

where:

- $K^{(m)}$ is the number of samples or batches,
- $|G|$ represents the total number of generators or components in the system,
- $\hat{y}_{k,d}^{(m)}$ is the predicted value of the k-th sample for the d-th component,
- $y_{k,d}^{(m)}$ is the true value for the same component.

The η^2 metric evaluates the relative error between predicted and true values, averaged across all samples and components in the system. A value of η^2 closer to 1 indicates that the model captures a large portion of the variance in the data, demonstrating high predictive accuracy. In this work, an η^2 value of 0.97055 indicates that the model explains over 97% of the variance in the predicted outputs, signifying excellent performance.

Chapter 3

Transfer Learning for Power Flow Problem

3.1 Methodology

This chapter involves two main phases: offline and online. First, the machine learning model is trained offline, where the required number of data points for transfer learning (TL) are determined, ensuring negative transfer is avoided. Once the model is trained, it is deployed online to adapt to changes in the power grid, where the feasibility of its performance is verified. The key advantage of this approach is that online computations are efficient, significantly speeding up power flow (PF) calculations.

In our approach, we utilize three distinct models to assess performance. The Source Model (SM) is trained on the source dataset, which is generated from the initial topology where all lines are connected. The Adapted Model (AM) is derived through transfer learning, applying knowledge from the source dataset to improve performance on the target dataset, where one line is cut off from the topology. Finally, the Target-Only Model (TOM) is trained from scratch, using only the target dataset, without any influence from the source dataset. This setup allows us to evaluate the effectiveness of transfer learning compared to training

solely on the target data and assess the benefits of adaptation in handling topology changes.

3.1.1 Source Model

The AC power flow model involves four primary variables at each bus: voltage magnitude, angle, and real and reactive power injections. For each bus, two out of these four variables are known, and the remaining two are deduced by solving a set of nonlinear power flow equations. Employing nodal analysis and considering constraints on power and/or voltage at bus k, the complex power is formulated as follows:

$$S_k = P_k + jQ_k = V_k I_k^* = V_k \sum_{n=1}^{N} Y_{kn} V_n^* \tag{3.1}$$

The specific calculations vary depending on whether the bus serves as a load or a generator:

- At a load bus (PQ) k the known variables are:

$$P_k = v_k \sum_{n=1}^{N} [G_{kn} \cos(\theta_k - \theta_n) + B_{kn} \sin(\theta_k - \theta_n)]$$
$$Q_k = v_k \sum_{n=1}^{N} [G_{kn} \sin(\theta_k - \theta_n) - B_{kn} \cos(\theta_k - \theta_n)]$$

- At a generator bus (PV) k:

$$P_k = v_k \sum_{n=1}^{N} [G_{kn} \cos(\theta_k - \theta_n) + B_{kn} \sin(\theta_k - \theta_n)]$$
$$v_k = v_k$$

Each $Y_{kn} = G_{kn} + jB_{kn}$ represents an element of the system admittance matrix, where v_k and v_n denote the voltage magnitudes at the respective buses, and θ_k/θ_n indicate their voltage angles.

To determine the voltage values at all bus locations efficiently, machine learning algorithms, specifically data-driven approaches, are utilized.

Data-driven solvers infer power flow (PF) solutions by mapping observed system inputs to outputs based on historical operational data. For PF problem:

$$[v_{\mathcal{L}}; \theta_{\mathcal{L},\mathcal{G}}] = f([P_{\mathcal{L}}; Q_{\mathcal{L}}; P_{\mathcal{G}}; v_{\mathcal{G}}])$$

Here, $P_{\mathcal{L}}, Q_{\mathcal{L}}$ represent power injections on load buses, and $P_{\mathcal{G}}, v_{\mathcal{G}}$ signify real power injections and voltage magnitudes on generation buses, respectively.

Considering multiple grid topologies, we define M different configurations, each differing by bus-branch connectivity. We assume all nodes (buses) remain connected across these topologies. A neural network (NN) can be parameterized for any topology m as $h^{(m)}(x_n, w)$, taking system inputs to produce outputs. The training dataset for any topology m, $T_m = \{(x_m^{(n)}, y_m^{(n)})\}_{n=1}^{\mathcal{N}_m}$, consists of input-output pairs where $\mathcal{N}_m$ is the number of training samples. The NN's parameters are optimized by minimizing:

$$J_{T_m}(w) = \frac{1}{\mathcal{N}_m} \sum_{n=1}^{\mathcal{N}_m} \left\| y_n^{(m)} - h^{(m)}(x_n^{(m)}, w) \right\|^2$$

This mean square error objective measures the accuracy of the NN predictions compared to the real solutions provided by a solver for each case. A significant limitation of current neural network (NN) based approaches is their dependency on a static system topology. The effectiveness of NN predictions hinges on the assumption that the underlying grid topology remains unchanged. However, as noted in recent studies [28], power grids are undergoing frequent topology reconfigurations to optimize for economic operation and reliability. These changes pose a challenge for traditional NN models, which are typically trained for a specific static configuration.

When the system topology is altered, the model ideally requires retraining from scratch to adapt to new conditions. This retraining process demands a considerable amount of new

training data and substantial computational time, which may not be feasible in operational settings. An alternative approach could involve training separate NNs for each potential system configuration. However, this method also demands extensive computational resources and presupposes a prior knowledge of all possible grid configurations; an impractical expectation given the unpredictable nature of power system operations, where unforeseen contingencies can occur.

Given these challenges, our approach for both PF leverages transfer learning techniques. Transfer learning allows our models to adapt to changes in topology with minimal need for data. By transferring learned features from one topology to another, we can significantly reduce the dependency on extensive retraining datasets and decrease computational demands. This methodology not only enhances the model's adaptability to unforeseen changes but also preserves its predictive accuracy under dynamic operational conditions.

3.1.2 Adapt to New Grid Topology

Data Regeneration

After a topology change, it becomes necessary to generate new data points. In this work, we propose a method for generating data that reflects the electrical interdependencies inherent in power system networks.

In power systems analysis, the power balance equation plays a crucial role, with power injection serving as the independent variable. This setup causes voltage levels (V) to depend directly on power injections. When the system's topology changes, V also changes, leading to mismatches between the expected model inputs and outputs. This necessitates training a model to adapt to new topologies. Generating these new datasets often involves setting specific power injections and employing the forward pass of the power balance equation via solvers.

$$Y = G + jB = \begin{bmatrix} Y_{1,1} & Y_{1,2} & \cdots & Y_{1,N} \\ Y_{2,1} & Y_{2,2} & \cdots & Y_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ Y_{N,1} & Y_{N,2} & \cdots & Y_{N,N} \end{bmatrix}$$

$$P_i + jQ_i = \sum_{i \in \mathcal{N}} V_i \left(V_i - V_j\right)^* Y_{ij}^*$$

$$P_{gi} - P_{di} = \sum_{j \in N} v_i v_j (G_{ij} \cos\theta_{ij} + B_{ij} \sin\theta_{ij}), \quad i \in \mathcal{N}$$

$$Q_{gi} - Q_{di} = \sum_{j \in N} v_i v_j (G_{ij} \sin\theta_{ij} - B_{ij} \cos\theta_{ij}), \quad i \in \mathcal{N}$$

The active power is denoted by P, the reactive power is denoted by Q, and the voltage magnitude by v. The voltage angle is represented by θ, while d and g refer to the power demand and generation, respectively. The admittance matrix is denoted by Y, with G and B representing the real and imaginary parts of the admittance matrix, respectively. Finally, N refers to the number of buses in the power system.

Traditional approaches to solving the PF and OPF equations, like the Newton-Raphson (NR) method, face challenges in such dynamic scenarios. The NR method iteratively adjusts the voltage estimates to balance power injections and withdrawals, but it can suffer from slow convergence rates. These drawbacks make the NR method less efficient for rapidly changing network configurations, highlighting the need for robust machine learning models.

Our innovative approach for generating datasets for new power flow (PF) scenarios involves fixing the voltage levels and recalculating power injections for all buses through the reverse pass of the power balance equation which is computationally easy. This method enables adaptation to changes in system topology without requiring extensive re-simulation for every possible configuration. Additionally, by utilizing the power balance equation, the solutions remain feasible and satisfy the constraints of power flow analysis.

Further elaboration of our approach extends to the bus admittance matrix, commonly

known as the Y matrix. The structure of the Y matrix is such that its diagonal elements reflect the total admittance connected to each bus, while the off-diagonal elements represent the negative sum of admittance between two specific buses. Additionally, the Y matrix is symmetric, meaning that $Y_{ij} = Y_{ji}$, which is a consequence of the reciprocal nature of the electrical connections between buses.

for example if the $N = 3$ the Y matrix is as below:

$$Y = \begin{bmatrix} y_1 + y_{12} + y_{13} & -y_{12} & -y_{13} \\ -y_{12} & y_2 + y_{12} + y_{23} & -y_{23} \\ -y_{13} & -y_{23} & y_3 + y_{13} + y_{23} \end{bmatrix}_{3\times3}$$

Consequently, disconnecting a branch affects four elements within the Y matrix. For example, if the branch between bus 1 and bus 2 is removed the canges are shown in red

$$Y = \begin{bmatrix} y_1 + y_{12} + y_{13} & -y_{12} & -y_{13} \\ -y_{12} & y_2 + y_{12} + y_{23} & -y_{23} \\ -y_{13} & -y_{23} & y_3 + y_{13} + y_{23} \end{bmatrix}_{3\times3}$$

Specifically, for the removal of a branch between buses k and l, the affected elements include off-diagonal elements $Y'_{kl} = Y'_{lk} = 0$ and diagonal elements $Y'_{kk} = Y_{kk} - Y_{kl}$, $Y'_{ll} = Y_{ll} - Y_{kl}$. The power injections S_k and S_l for these buses must be recalculated. The expansion of Equation 3.1 is as follows:

$$\begin{bmatrix} S_1 \\ \vdots \\ S_k \\ \vdots \\ S_l \\ \vdots \\ S_N \end{bmatrix} = \begin{bmatrix} V_1 \\ \vdots \\ V_N \end{bmatrix} \odot \begin{bmatrix} V_1 & \cdots & V_N \end{bmatrix}^* \times \begin{bmatrix} Y_{1,1} & \cdots & Y_{1,k} & \cdots & Y_{1,l} & \cdots & Y_{1,N} \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ Y_{k,1} & \cdots & Y_{k,k} & \cdots & Y_{k,l} & \cdots & Y_{k,N} \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ Y_{l,1} & \cdots & Y_{l,k} & \cdots & Y_{l,l} & \cdots & Y_{l,N} \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ Y_{N,1} & \cdots & Y_{N,k} & \cdots & Y_{N,l} & \cdots & Y_{N,N} \end{bmatrix}^*$$

Thus, updating the dataset involves recalculating the power injections S_l and S_k for the affected buses. Consequently, not only is the reverse pass computationally straightforward to solve, but it is also necessary to recalculate it for only two of the N available buses that are adjacent to any changes rather than all buses.

Avoiding Negative Transfer

In [60], the authors introduce a Rank of Domain (ROD) metric to determine which dataset(s) will yield the best performance in a target domain when used as source domains. This metric leverages KL divergence to measure differences between data distributions. A pair of domains with smaller KL divergence values are more likely to transfer knowledge effectively. Empirically, when this metric is used to rank datasets as potential source domains, the rankings align well with actual performance improvements in the target domain.

Additionally, the mean squared error (MSE) of the model's predictions on new data is an indicator of transferability. A higher MSE signals a greater discrepancy between learned representations and the new task, necessitating more labeled data. Since poor model performance on unseen samples suggests a substantial distribution shift, the prediction error directly informs the degree of adaptation required.

Considering three key factors—distribution divergence, algorithm selection, and the size of labeled data—we aim to systematically avoid negative transfer. To achieve this, we propose

that the number of required data points can be guided by both the divergence between distributions and the prediction MSE of the model on new topology data. Keeping this in mind, we can measure the divergence between modified topologies and the initial one to determine the optimal number of data points required for effective transfer.

Next, we assess the occurrence of negative transfer (NT) by checking whether NTG 2.28 is positive, confirming NT's presence, or negative, indicating successful transfer. The combined assessment of distribution divergence and prediction error enables a more comprehensive strategy to mitigate NT, ensuring that the adapted model retains the benefits of transferred knowledge without performance degradation.

First, we begin by calculating the KL divergence. The formula for the KL divergence of a discrete probability distribution P relative to a distribution Q over the same space is given by:

$$D_{KL}(\mathcal{P} \parallel \mathcal{Q}) = \sum_{x} \mathcal{P}(x) \log \left(\frac{\mathcal{P}(x)}{\mathcal{Q}(x)} \right)$$

If the variables are independent:

$$\mathrm{KL}(\mathcal{P} \parallel \mathcal{Q}) = \sum_{i} \mathrm{KL}(\mathcal{P}_i \parallel \mathcal{Q}_i) \quad \text{(over all } S_i\text{)}$$

$$\mathrm{KL}(\mathcal{P}(S_1, \ldots, S_N) \parallel \mathcal{Q}(S_1, \ldots, S_N)) = \sum_{i=1}^{N} \mathrm{KL}(\mathcal{P}(S_i) \parallel \mathcal{Q}(S_i))$$

$$\mathrm{KL}(\mathcal{P}(S_1) \parallel \mathcal{Q}(S_1)) + \cdots + \mathrm{KL}(\mathcal{P}(S_N) \parallel \mathcal{Q}(S_N))$$

Just two KL terms will remain. The rest are equal to zero (for buses k and l):

$$\mathrm{KL}(\mathcal{P}(S_k) \parallel \mathcal{Q}(S_k)) + \mathrm{KL}(\mathcal{P}(S_l) \parallel \mathcal{Q}(S_l))$$

Therefore, the KLD will be calculated for buses adjacent to any changes in the network, rather than for all buses.

$$= \sum_x \mathcal{P}(S_k) \log\left(\frac{\mathcal{P}(S_k)}{\mathcal{Q}(S_k)}\right) + \sum_x \mathcal{P}(S_l) \log\left(\frac{\mathcal{P}(S_l)}{\mathcal{Q}(S_l)}\right) \quad \text{(over all points)}$$

To find the worst case:

$$\max\left(\log\left(\frac{\mathcal{P}(S_k)}{\mathcal{Q}(S_k)}\right)\right) = M_k$$

$$\max\left(\log\left(\frac{\mathcal{P}(S_l)}{\mathcal{Q}(S_l)}\right)\right) = M_l$$

$$= \sum_x \mathcal{P}(S_k) M_k + \sum_x \mathcal{P}(S_l) M_l$$

$$= M_k \times \sum_x \mathcal{P}(S_k) + M_l \times \sum_x \mathcal{P}(S_l)$$

In probability theory, $\sum_x \mathcal{P}(S) = 1$ because the sum of probabilities for all possible outcomes in a probability distribution must equal 1, representing the certainty that some outcome will occur. Hence:

$$= M_k + M_l$$

Now, to calculate $\frac{\mathcal{P}(S_k)}{\mathcal{Q}(S_k)}$, We have following equation in [61]:

$$\mathcal{P}_y = \mathcal{P}_x \left|\frac{dx}{dy}\right|$$

In our case, since the voltage is considered fixed, we have:

$$\mathcal{Q}_V = \mathcal{Q}_S \left|\frac{dS}{dV}\right| = \mathcal{P}_S \left|\frac{dS'}{dV}\right|$$

Therefore:

$$\frac{\mathcal{P}_S}{\mathcal{Q}_S} = \left|\frac{dS}{dV}\right| \times \frac{1}{\left|\frac{dS'}{dV}\right|}$$

Here is the gradient of S with respect to V:

$$\frac{dS}{dV} = \begin{bmatrix} \overline{VY_1}^* + V_1^* Y_{11}^* & V_1 V_2^* & \cdots & V_1 Y_{n1}^* \\ V_2 Y_{12}^* & \overline{VY_2}^* + V_2 Y_{22}^* & \cdots & V_2 Y_{n2}^* \\ \vdots & \vdots & \ddots & \vdots \\ V_n Y_{1n}^* & V_n V_{2n}^* & \cdots & \overline{VY_n}^* + V_n Y_{nn}^* \end{bmatrix}$$

Only two buses are affected, so we will examine the changes in those two rows separately. Currently, we are analyzing the change in distribution for bus k. Since we are dealing with a single row rather than a matrix, we will use the norm instead of the determinant.

$$\|\frac{\partial S_k}{\partial V}\|_2 = \sqrt{|\sum_{j=1}^{N} \left(V_j^* Y_{kj}^*\right) + V_k Y_{kk}^*|^2 + \sum_{\substack{j=1 \\ j\neq k}}^{N} \left|V_k Y_{kj}^*\right|^2}$$

When the topology changes, the alterations are highlighted in red:

$$\frac{\left\|\frac{\partial S_k}{\partial V}\right\|_2}{\left\|\frac{\partial S'_k}{\partial V}\right\|_2} = \sqrt{\frac{\left|\sum_{j=1}^{N} \left(V_j^* Y_{kj}'^*\right) + V_k Y_{kk}'^* + \textcolor{red}{V_l^* Y_{kl}^* + V_k Y_{kl}^*}\right|^2 + \sum_{\substack{j=1 \\ j\neq k}}^{N} \left|V_k Y_{kj}'^*\right|^2 + \textcolor{red}{|V_k Y_{kl}^*|^2}}{\left|\sum_{j=1}^{N} \left(V_j^* Y_{kj}'^*\right) + V_k Y_{kk}'^*\right|^2 + \sum_{\substack{j=1 \\ j\neq k}}^{N} \left|V_k Y_{kj}'^*\right|^2}}$$

Hence:

$$M_k = \max\left(\log\left(\frac{\left|\frac{\partial S_k}{\partial V}\right|}{\left|\frac{\partial S'_k}{\partial V}\right|}\right)\right)$$

$$= \max\left(\log\left(\sqrt{\frac{\left|\sum_{j=1}^{N} \left(V_j^* Y_{kj}'^*\right) + V_k Y_{kk}'^* + \textcolor{red}{V_l^* Y_{kl}^* + V_k Y_{kl}^*}\right|^2 + \sum_{\substack{j=1 \\ j\neq k}}^{N} \left|V_k Y_{kj}'^*\right|^2 + \textcolor{red}{|V_k Y_{kl}^*|^2}}{\left|\sum_{j=1}^{N} \left(V_j^* Y_{kj}'^*\right) + V_k Y_{kk}'^*\right|^2 + \sum_{\substack{j=1 \\ j\neq k}}^{N} \left|V_k Y_{kj}'^*\right|^2}}\right)\right)$$

and

$$M_l = \max\left(\log\left(\frac{\left|\frac{\partial S_l}{\partial V}\right|}{\left|\frac{\partial S'_l}{\partial V}\right|}\right)\right)$$

$$= \max\left(\log\left(\sqrt{\frac{\left|\sum_{j=1}^{N}\left(V_j^* Y_{lj}'^*\right) + V_l Y_{ll}'^* + V_k^* Y_{lk}^* + V_l Y_{lk}^*\right|^2 + \sum_{\substack{j=1 \\ j\neq l}}^{N}\left|V_l Y_{lj}'^*\right|^2 + \left|V_l Y_{lk}^*\right|^2}{\left|\sum_{j=1}^{N}\left(V_j^* Y_{lj}'^*\right) + V_l Y_{ll}'^*\right|^2 + \sum_{\substack{j=1 \\ j\neq l}}^{N}\left|V_l Y_{lj}'^*\right|^2}}\right)\right)$$

3.1.3 Algorithm

Algorithm 1 Offline Training

Input: Training dataset $\mathcal{S} = \{(x_s^i, y_s^i)\}_{i=1}^{n_s}$, Step sizes α, Number of epochs E **Output:** $\mathbf{w}_{SM}$: Optimal parameter for Source Model (SM)

1: Initialize $\mathbf{w}_{\text{SM}}$ randomly
2: **for** epoch = 1 to E **do**
3: Sample batch of points from $\mathcal{S}$
4: Evaluate $\nabla J_{\mathcal{S}}(\mathbf{w}_{\text{SM}})$ using $\mathcal{S}$
5: Update source model parameters with gradient descent:

$$\mathbf{w}_{SM} = \mathbf{w}_{SM} - \alpha \nabla J_{\mathcal{S}}(\mathbf{w}_{SM})$$

6: **end for**

With the number of required data points identified to compensate for errors and avoid negative transfer, we can now fine-tune the source model and achieve an adaptive model.

Algorithm 2 Online Training

Input: $\mathbf{w}_{\text{SM}}, \gamma, N_{\text{TL}}$: Required number of data points for Transfer Learning (TL)
Output: $\mathbf{w}_{\text{AM}}$: Adapted parameters for new configuration

1: Initialize $\mathbf{w}_{\text{AM}} = \mathbf{w}_{\text{SM}}$: Use the source model's parameters as the initial knowledge
2: **while** system in operation **do**
3: Generate $\mathbf{N}_{\text{TL}}$ labeled target set $\mathcal{T}_{m,l} = \{(x_{m,l}^j, y_{m,l}^j)\}_{j=1}^{N_{\text{TL}}}$ using Reverse Pass
4: Compute the adapted parameters with gradient descent:

$$\mathbf{w}_{\text{AM}} = \mathbf{w}_{\text{AM}} - \gamma \nabla J_{\mathcal{T}_{m,l}}(\mathbf{w}_{\text{AM}})$$

5: **end while**

3.2 Experiments

This section outlines the computational environment, datasets, and methodology employed in this research. It begins by describing the hardware and software configurations, followed by an overview of the datasets used for training, validation, and testing. The subsequent sections discuss source model selection, hyperparameter tuning, and comparative evaluations. Additionally, it explains the process of generating new data points for the second topology while ensuring feasibility conditions, selecting the appropriate amount of data, and implementing strategies to prevent negative transfer learning. Finally, the section explores how transfer learning compensates for changes in topology.

3.2.1 Environment and Datasets

The experiments were conducted on a MacBook Pro equipped with an Apple M2 processor and 16 GB of RAM. The learning framework was developed in Python 3.9 using TensorFlow 2.13.0 and Keras 2.13.1. The same computational environment is maintained across Chapter 4 for consistency.

To evaluate the proposed algorithm, experiments were performed on three well-established benchmark systems: IEEE 14-bus, IEEE 118-bus, and PEGASE 1354-bus systems (used to assess scalability). The dataset was generated using MATPOWER [62]. System variations were introduced by randomly adjusting nodal active and reactive power demands within 80% to 120% of their nominal values. This approach aligns with standard practices in the literature [63, 64, 65].

The ±20% variation (i.e., 80% to 120% of nominal values) is commonly used to simulate realistic fluctuations in power demand, as observed in real-world systems due to consumer behavior, environmental conditions, and time-of-day effects. This range captures both peak load (up to 120%) and off-peak demand (down to 80%). It also allows researchers to test model robustness under typical and extreme, yet feasible, operating conditions. Additionally,

it implicitly accounts for measurement noise, which is often reflected in the data distribution and mitigated through preprocessing to improve model reliability and generalization.

In MATLAB, the approach involved perturbing default cost function coefficients to modify generation costs and randomly sampling power generation, followed by the 'runpf' function. For small-scale systems, direct sampling of power generation parameters followed by the 'runpf' function was feasible. However, this approach proved ineffective for larger systems due to frequent infeasibilities where the total generated power did not match the demand, leading to solver failures. To mitigate this, the strategy was adjusted to perturb the cost function coefficients and utilize the 'runopf' function, which proved more robust for large systems (1354-bus system).

The dataset for each system (the IEEE 14-bus, 118-bus, and PEGASE 1354-bus system) comprises 20,000 feasible scenarios, emulating real datasets containing feasible load flow data recorded by system operators. The data was then partitioned into three subsets: 80% for training, 10% for validation, and 10% for testing, ensuring a balanced distribution for effective model evaluation.

Given the inherent uncertainties associated with renewable energy sources, we assume that each non-generator bus is equipped with either a wind farm or a solar photovoltaic (PV) station. In 2020, wind and solar energy contributed approximately 8.4% and 2.3% of the total power generation in the United States, respectively. To align with these proportions, the deployment and parameter selection of distributed energy resources (DERs) in this study reflect these percentages.

The IEEE 118-bus system includes 64 load buses, with 24 designated for wind farms and the remaining 40 for solar PV installations. Each wind farm is modeled to have a single turbine with a swept area of 1,800 m^2, while each solar farm consists of 10,000 panels, each measuring 1.67 m^2. Uncertainties in the wind and solar generation are modeled using Weibull distributions. These renewable sources are integrated into the power flow equations, influencing power injections [66].

3.2.2 Source Model

The source model designed for the power flow problem is a fully connected feedforward neural network (FNN). This model aims to predict bus voltages by learning from simulated power system data. The architecture consists of an input layer, one hidden dense layer, and an output layer, as described below.

Model Architecture

The model is a multi-layer perceptron designed to process power injection data and predict bus voltage magnitudes and angles. The model takes as input a feature vector representing the system state, including the power injection values at load buses, active power injections, and voltage magnitudes at generator buses. The output consists of voltage magnitudes at load buses and voltage angles for all buses.

To determine the optimal model architecture, an extensive grid search was conducted to explore various hyperparameters, including the number of layers, neurons per layer, activation functions, learning rates, batch sizes, and optimization algorithms. The architecture ultimately selected demonstrated a good balance between performance and generalization, and was inspired by successful applications in prior studies involving similar power system datasets.

Specifically, the model depth was varied between 1 to 4 layers, with each layer tested using 64 to 512 neurons. Activation functions such as ReLU, Leaky ReLU, Sigmoid, Tanh, and ELU were evaluated, with ReLU emerging as the most effective due to its simplicity and strong empirical performance across tasks. Learning rates ranging from 10^{-2} to 10^{-5} and batch sizes from 32 to 512 were also tested. Among the optimizers considered—Adam, AdamW, SGD, and RMSprop—Adam provided the best convergence behavior. To avoid overfitting on the source domain, techniques such as early stopping based on validation loss and dropout regularization were employed during training. This tuning process ensured that the selected architecture was both data-efficient and well-suited to transfer to the target domain.

The input layer's dimensionality corresponds to the number of features in the dataset, which is dynamically set to the number of input variables (n_{features}) in the training data.

The architecture includes:

- **Input Layer:** A dense layer with dimensionality equal to the number of features in the dataset, n_{features}.
- **Hidden Layer:** A fully connected layer with a case-dependent number of units, using the Rectified Linear Unit (ReLU) activation function:

$$\text{ReLU}(x) = \max(0, x)$$

  This activation function introduces non-linearity into the model, which is crucial for capturing the inherent non-linear behavior of power flow equations.
- **Output Layer:** A dense layer with dimensionality equal to the number of target variables (n_{targets}). This layer has no activation function, which is appropriate for regression tasks where continuous values such as bus voltages and angles are predicted.

Table 3.1: Model Parameters for Different Cases

Case	Input Layer Units	Hidden Layer Units	Output Layer Units	Total Parameters
14	26	256	22	12,566
118	234	1024	181	426,165
1354	2708	4024	2447	20,742,143

All parameters are trainable, ensuring the model can effectively learn the complex mappings between input and output variables.

Preprocessing

As a preprocessing step, it was essential to scale the data into appropriate ranges to ensure effective training. The best results were achieved by transforming the voltage angles from degrees to radians, as this transformation brought the voltage angles into the same range as

the voltage magnitudes. Additionally, the power injections were normalized by dividing them by 100, corresponding to the base MVA of the system. These preprocessing steps improved the model's ability to converge by ensuring that the input data was well-scaled and within a similar range. Furthermore, branches that caused disconnection in the network were removed to maintain the structural integrity of the system. Additionally, the slack bus voltage angle was eliminated from the output since it consisted of an entirely zero column.

Loss Function and Optimization

The model is trained using the Mean Squared Error (MSE) loss function, which is defined as:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{\mathcal{N}} \sum_{i=1}^{\mathcal{N}} (y_i - \hat{y}_i)^2$$

where $\mathcal{N}$ is the number of samples, y_i is the true value, and $\hat{y}_i$ is the model's predicted value. The optimizer used is Adam, a variant of stochastic gradient descent, with a learning rate of 1×10^{-4}. Adam was chosen for its efficient handling of sparse gradients and ability to adapt learning rates during training.

Training Procedure

The model is trained using 5-fold cross-validation to ensure robust generalization to unseen data. In each fold, the data is split into training and validation sets, and the model is trained for 200 or 100 epochs with a batch size of 20. Early stopping is applied to prevent overfitting, with a patience value of 10 epochs. This means that training stops if the validation loss does not improve for 10 consecutive epochs, and the best-performing model is restored based on validation loss.

To further ensure robustness, the model is trained multiple times within each fold, with different random seeds initialized at the beginning of each run. This helps to avoid the model converging to a suboptimal solution due to poor initialization. The model with the best

validation loss is saved after each fold for further evaluation.

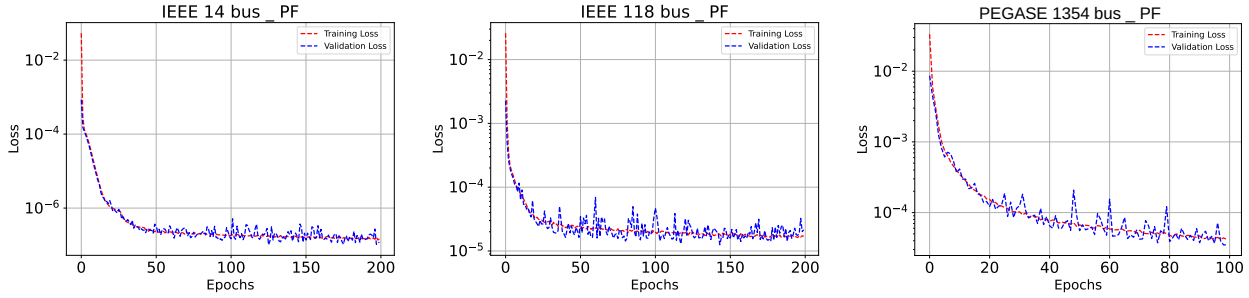
Results and Performance

The performance of the source model is evaluated using the validation loss from each fold, and the average loss across all folds is computed.

Table 3.2: Source Model Performance Metrics for Different Cases

Metric	Case 14	Case 118	Case 1354
Mean Validation MSE	1.33×10^{-7}	1.13×10^{-5}	3.55×10^{-5}
Best Validation Loss	8.46×10^{-8}	6.80×10^{-6}	3.22×10^{-5}
RMSE	0.00035	0.0036	0.0057
MAPE	1.78%	0.53%	5.05%
95% Confidence Interval	$(1.23 \times 10^{-7}, 1.42 \times 10^{-7})$	$(9.97 \times 10^{-6}, 1.27 \times 10^{-5})$	$(3.47 \times 10^{-5}, 3.63 \times 10^{-5})$
η^2	0.98	0.99	0.95
R^2 Score	1.00	1.00	1.00
MSE (Voltage Magnitude at Load Buses)	1.2×10^{-7}	1.44×10^{-5}	3.58×10^{-5}
MSE (Voltage Angle)	1.4×10^{-7}	1.21×10^{-5}	2.86×10^{-5}

The results show promising initial performance, with the neural network providing comparable results to classical methods but with significantly reduced computation time, which is crucial for real-time applications.



(a) SM for IEEE 14-bus (b) SM for IEEE 118-Bus (c) SM for PEGASE 1354-bus

Figure 3.1: Training and validation loss over the epochs for the power flow prediction model. Both training and validation loss decrease steadily and converge after approximately 60 epochs, indicating good generalization and minimal overfitting. The loss is plotted on a logarithmic scale to highlight the significant reduction in loss.

3.2.3 Adapt to New Grid Topology

To adapt to a new topology, multiple steps are taken. First, the required amount of data is determined while avoiding negative transfer, considering the distributional distance between

new and old data points and the source model's performance on the new data. Next, new data points are generated using the reverse pass process. Finally, the adapted model is trained.

Required number of data points for transfer learning: $N_{\mathbf{TL}}$

To determine the required number of data points for transfer learning while preventing negative transfer (where training fails and results in an error), a subset of branches was randomly selected. A small initial set of data points was generated for these branches, and additional data points were incrementally added until both the MSE and KLD fell below a predefined threshold. Ultimately, the number of data points needed for each new topology was determined while ensuring that MSE and KLD remained low and negative transfer never occurred. The experiment demonstrates that, on average, only 3% of the source data is required.

Data Regeneration

To regenerate data, the reverse pass technique is applied to 3% of the source data points. During this process, the voltage of all buses remains fixed, leading to adjustments in power injection values. Specifically, the active and reactive power injections at load buses and the active power injection at generation buses are modified in the input matrix. The new inputs are as below:

$$[v_{\mathcal{L}}; \theta_{\mathcal{L},\mathcal{G}}] = f([P_{\mathcal{L}-\text{new}}; Q_{\mathcal{L}-\text{new}}; P_{\mathcal{G}-\text{new}}; v_{\mathcal{G}}])$$

This ensures that the regenerated data maintains consistency with the underlying system constraints while introducing variations in power injections.

Adapted Model

The Adapted Model (AM) retains the same architecture as the pre-trained source model (SM), which is loaded from a previously trained Keras model. Instead of being trained from scratch,

AM leverages transfer learning by fine-tuning SM on a new target dataset. The model is trained for 15 epochs with a batch size of 5, using a designated training set, while a separate validation set is used to monitor generalization performance. Since SM has already learned useful representations from its original task, AM benefits from knowledge transfer, leading to faster convergence and improved accuracy. After training, predictions are generated on an evaluation dataset to assess the model's performance. This approach significantly reduces training time and enhances performance, as demonstrated in the comparative results with the target-only model (TOM).

Results and Performance

Table 3.3 presents a comparative analysis of the Adapted Model (AM) and the Target-Only Model (TOM) across three different cases: Case 14, Case 118, and Case 1354. The performance is evaluated using key metrics such as Mean Squared Error (MSE), Mean Absolute Percentage Error (MAPE), R^2 score, η^2 score, and training time. The results highlight the effectiveness of transfer learning, as detailed below.

Mean Squared Error (MSE): The AM achieves significantly lower MSE values compared to the TOM in all cases, indicating superior prediction accuracy. In Case 14, the AM attains an MSE of 4.00×10^{-7}, while the TOM exhibits a much higher error of 0.04. Similarly, in Case 1354, the AM maintains an MSE of 1.7×10^{-5}, whereas the TOM records 0.067. This substantial reduction in error demonstrates the benefits of knowledge transfer from the source model.

Mean Absolute Percentage Error (MAPE): The adapted model consistently achieves significantly lower MAPE values, highlighting its robustness. For instance, in Case 14, the AM's MAPE is 3.59%, whereas the TOM reaches 263.87%, indicating severe instability. A similar trend is observed in Case 1354, where the AM maintains a reasonable 4.16%, whereas the TOM's error soars to 339.34%. This emphasizes the AM's enhanced generalization

capability.

R^2 Score: The R^2 score, which measures the proportion of variance explained by the model, further supports the superiority of the AM. Across all cases, the AM maintains a near-perfect score of 0.99, whereas the TOM struggles, particularly in Case 118, where its score drops to 0.56. This suggests that the TOM fails to capture the underlying patterns effectively when trained without prior knowledge.

η^2 Score: The η^2 score, which quantifies the impact of knowledge transfer, is consistently high for the AM, with values exceeding 0.96 in all cases. Conversely, the TOM exhibits negative values in Case 14 (-1.64) and Case 1354 (-2.39), signifying a lack of effective learning when trained from scratch. These results reaffirm the substantial benefits of transfer learning in improving model performance.

Training Time: A significant advantage of the AM is its computational efficiency. As observed, the training time for the AM is drastically lower than that of the TOM. For Case 14, the AM requires only 1.94 seconds, whereas the TOM takes 17 minutes. This trend persists in Case 1354, where the AM completes training in 8.91 seconds, compared to 40 minutes for the TOM. The substantial reduction in training time indicates that transfer learning not only enhances accuracy but also accelerates convergence.

Conclusion: The results in Table 4.3 strongly support the effectiveness of transfer learning. The AM consistently outperforms the TOM across all evaluation metrics, achieving lower error rates, higher accuracy, and significantly reduced training times. These findings highlight the advantages of incorporating prior knowledge into the learning process, demonstrating the superior efficiency and robustness of the adapted model in various scenarios.

For example, consider a scenario where the branch connecting Bus 4 and Bus 5 in IEEE 118-bus system is removed.

Table 3.3: Performance Metrics for AM and TOM Models Across Different Cases

Metric	**Case 14**		**Case 118**		**Case 1354**	
	AM	**TOM**	**AM**	**TOM**	**AM**	**TOM**
MSE	4.00×10^{-7}	0.04	3.58×10^{-6}	0.03	1.7×10^{-5}	0.067
MAPE	3.59%	263.87%	1.02%	45.98%	4.16%	339.34%
R^2 **Score**	0.99	0.86	0.99	0.56	0.99	0.86
η^2	0.96	-1.64	0.99	0.54	0.96	-2.39
Training Time	1.94 S	17 min	2.45 S	20 min	8.91 S	40 min

In dynamic power systems, critical violations require rapid response and adaptation to maintain system stability. Training from scratch, while accurate, is computationally intensive and slow to adjust to changes in system topology. For instance, when training TOM over 10 epochs using all data points, the process takes approximately 17 hours and achieves a mean squared error (MSE) of 2×10^{-4} with an R^2 score of 0.98. Even with only 2% of the data, the time drops to 1 hour, but the performance degrades significantly, yielding an MSE of 1.43×10^{-2} and an R^2 score of just 0.13. In contrast, the Adapted Model (AM) delivers significantly better results: over the same number of epochs, the entire adaptation process—including new data generation, model retraining, and prediction—completes in just 1.75 seconds, achieving an MSE of 4.9×10^{-5} and an R^2 score of 0.99. These results highlight the superiority of AM in both speed and predictive accuracy, making it highly suitable for real-time response in evolving grid environments.

The resulting training and validation losses for this case are illustrated in Figure 3.2. In this figure, the y-axis is plotted on a logarithmic scale for clarity. It compares the performance of two models: the adapted model (which uses transfer learning from a source model) and the target-only model (trained from scratch on the target data). From this figure, we can make several observations:

Adapted Model vs. Target-Only Model: The adapted model consistently achieves a lower loss than the target-only model throughout training.

Lower Initial Error: The adapted model starts with a much smaller initial error, thanks to the knowledge transferred from the source model.

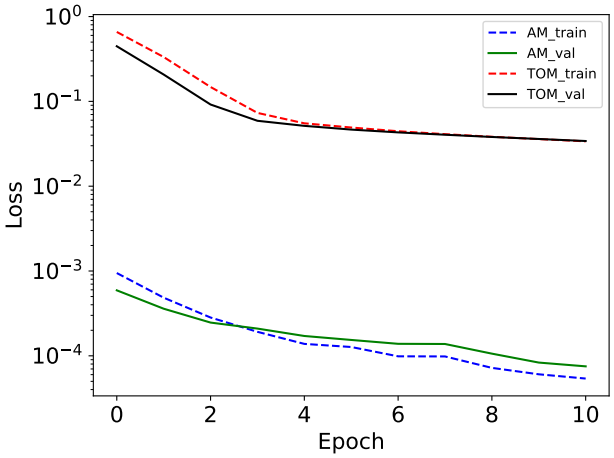


Figure 3.2: Loss Gap Between Adapted Model and Target-Only Model

Significant Performance Gap: There is a significant gap between the loss curves of the two models. The adapted model's loss remains well below that of the target-only model at each training step.

Impact of Transfer Learning: This considerable gap highlights the effectiveness of transfer learning. The adapted model's ability to exploit prior knowledge results in a strong improvement over the target-only model, demonstrating promising results in this scenario.

Overall, the removal of the branch between Bus 4 and Bus 5 leads to a scenario where the transfer-learned (adapted) model vastly outperforms the model trained solely on target data. This underscores the significant benefits of transfer learning, as the adapted model not only begins with a lower error but also maintains superior performance throughout the training process.

3.2.4 Comparison

Table 3.4 presents a comparative analysis of TL with existing methods for power flow analysis, focusing on their respective limitations and the advantages introduced by TL.

The **Machine Learning-Based ACPF Initialization** approach proposed by [29] aims to enhance AC power flow (ACPF) computation efficiency. However, it exhibits several limitations:

- **Limited time improvement:** The method reduces ACPF computation time by only **24% for small networks** and **64% for large networks**, which remains suboptimal for large-scale power systems.
- **Dependency on Newton-Raphson (NR):** Despite leveraging machine learning, the approach still relies on the conventional **Newton-Raphson method** for solving power flow equations, thereby inheriting its iterative nature and convergence constraints.
- **Topology dependence:** The model requires explicit **branch status encoding**, such as one-hot encoding for N-1 contingency scenarios.

In contrast, **Transfer Learning (TL)** addresses these shortcomings through the following key advantages:

- **Superior speed-up:** TL achieves significant computational efficiency, with **525.77× speed-up (Case 14)**, **489.80× (Case 118)**, and **269.36× (Case 1354)**, reducing the overall training time by more than **99.6%** across different cases.
- **Elimination of Newton-Raphson dependency:** TL completely removes reliance on iterative solvers, thus avoiding convergence issues associated with NR-based approaches.
- **Topology-free input:** Unlike its predecessor, TL does not require **branch status encoding** for network contingencies, significantly enhancing its generalization capability.

The **Typed Graph Neural Networks (TGN) for Power Flow** model, introduced by [32], applies graph-based learning techniques to power system analysis. However, it suffers from the following challenges:

- **High complexity:** The approach necessitates multiple interconnected components, including **feature encoders, message-passing layers, update functions, and decoders**, resulting in increased model intricacy.

- **Data inefficiency:** The model requires **large-scale labeled datasets** to generalize effectively, making it impractical in scenarios where data availability is limited.
- **Difficult integration:** The deployment of TGN in practical power systems is constrained by its reliance on **specialized graph-based machine learning frameworks**, complicating real-world implementation.

Conversely, **Transfer Learning (TL)** offers several advantages over TGN-based approaches:

- **Simplified model architecture:** TL employs a **fully connected neural network (FCNN)**, eliminating the need for graph convolutions and iterative update mechanisms.
- **Higher data efficiency:** TL requires significantly fewer labeled samples, making it **more suitable for real-world grid applications**, where acquiring large labeled datasets is costly.
- **Seamless integration:** Unlike TGN, TL integrates efficiently into **existing machine learning pipelines** without necessitating additional specialized frameworks.

Overall, the findings indicate that **Transfer Learning** provides a computationally efficient, data-efficient, and easily deployable solution compared to traditional machine learning-based ACPF initialization and graph-based power flow analysis methods.

Table 3.4: Comparison of TL with Existing Approaches in PF Analysis

Approach	Limitations	Advantages of Transfer Learning
ML-Based ACPF Init. (Dong et al., 2022) [29]	• **Limited time improvement:** Reduces computation time by 64%. • **Dependency on Newton-Raphson:** Still relies on traditional solvers. • **Topology-dependent:** Requires branch status encoding.	• **Superior speed-up:** Reduces computation time by 99.6%. • **No Newton-Raphson dependency:** Eliminates reliance on iterative solvers. • **Topology-free input:** Does not require branch status encoding.
TGN for Power Flow (Lopez et al., 2023) [32]	• **High complexity:** Requires multiple learning components. • **Data inefficiency:** Needs large-scale labeled datasets. • **Difficult integration:** Requires a specialized ML framework.	• **Simplified model:** Uses a fully connected neural network. • **More data-efficient:** Requires fewer labeled samples. • **Easier deployment:** Integrates seamlessly into existing ML pipelines.

Chapter 4

Transfer Learning for Optimal Power Flow problem

4.1 Methodology

This chapter follows the same two-phase approach as Chapter 3: offline training and online adaptation. The machine learning model is first trained offline, determining the necessary data points for transfer learning (TL) while avoiding negative transfer. Once trained, the model is deployed online to adapt to power grid changes efficiently, accelerating optimal power flow (OPF) calculations.

To evaluate performance, we compare three models. The Source Model (SM) is trained on a fully connected topology. The Adapted Model (AM) applies transfer learning from SM to a modified topology where a line is removed. The Target-Only Model (TOM) is trained solely on the target dataset without prior knowledge. This setup quantifies the benefits of transfer learning in handling topology changes.

4.1.1 Source Model

The optimal power flow (OPF) problem aims to determine the most cost-effective generation dispatch under specified load conditions, adhering to constraints imposed by power flow equations as well as limits on generation and voltage. The formulation of the OPF problem is given by:

$$\min_{P_g^{(m)}, Q_g^{(m)}, V^{(m)}} \sum_{i \in \mathcal{G}} C_i(P_{g_i}^{(m)}) \tag{4.1}$$

with constraints:

$$P_{g_i}^{min} \leq P_{g_i}^{(m)} \leq P_{g_i}^{max}, \quad \forall i \in \mathcal{G} \tag{4.2}$$

$$Q_{g_i}^{min} \leq Q_{g_i}^{(m)} \leq Q_{g_i}^{max}, \quad \forall i \in \mathcal{G} \tag{4.3}$$

$$V_i^{min} \leq V_i^{(m)} \leq V_i^{max}, \quad \forall i \in \mathcal{N} \tag{4.4}$$

Here, $C_i(\cdot)$ represents the cost of generating power at bus $i \in \mathcal{G}$. The terms $P_{g_i}^{max}$ ($P_{g_i}^{min}$), $Q_{g_i}^{max}$ ($Q_{g_i}^{min}$), and V_i^{max} (V_i^{min}) specify the upper (lower) bounds for real and reactive power generation and the voltage limits at node i, respectively.

Data-driven solvers for the optimal power flow (OPF) problems deduce optimal settings by mapping specified system demands to outputs, utilizing historical operational data. For instance:

$$[v; \theta_{\mathcal{L},\mathcal{G}}] = f([P_d; Q_d])$$

In this formulation, P_d and Q_d represent the real and reactive power demands across all buses, respectively. The model outputs, v and $\theta_{\mathcal{L},\mathcal{G}}$, are the optimized voltage magnitudes at each bus and angles at load and genaration buses, ensuring efficient and stable power distribution throughout the network.

We consider M different grid configurations, each with varying bus-branch connectivity while ensuring all buses remain connected. A neural network (NN) is parameterized for any

topology m as $h^{(m)}(x_n, w)$ and trained on dataset $T_m = \{(x_m^{(n)}, y_m^{(n)})\}_{n=1}^{\mathcal{N}_m}$ by minimizing the mean squared error loss.

Traditional NN-based methods assume a static topology, but power grids frequently reconfigure for efficiency and reliability [28], making such models impractical. Retraining for each topology is computationally expensive and infeasible.

To address this, we leverage transfer learning, enabling models to adapt to topology changes with minimal data and computation. This approach reduces retraining needs while preserving predictive accuracy in dynamic grid conditions.

4.1.2 Adapt to New Grid Topology

Data Regeneration

A similar strategy is employed for optimal power flow problems as 3.1.2, with a key difference: with voltage, power generation at each bus are also fixed. When the network changes, the existing power generation may lead to infeasibility. To maintain feasibility, we adjust the power demand accordingly, ensuring the power balance equation remains satisfied. Consequently, the power demand adjusts accordingly. By using the reverse pass of the power balance equation, we can efficiently regenerate the power injections for all buses. Since the voltage and power generation are both fixed, this process ensures that the OPF objectives (referenced as objective 4.1) and all associated constraints (constraints 4.2, 4.3, and 4.4) are satisfied, even under new operating conditions. This method streamlines the adaptation of power systems to evolving demands and topologies, significantly enhancing the flexibility and responsiveness of grid management strategies.

An exception applies to generation buses without load. If there is no load and the demand is zero, power injection is equal to power generation. In this case, power generation will adjust accordingly, eliminating the need for model fine-tuning. Instead, the focus shifts to monitoring cost variations. The experiment will demonstrate that such buses are rare, and the cost changes in these cases are minimal.

Avoiding Negative Transfer

Following the approach in Chapter 3, we systematically mitigate negative transfer (NT) by considering distribution divergence, algorithm selection, and labeled data availability. The required data points are determined based on distribution divergence and the model's prediction MSE on new topologies.

Negative transfer is assessed by evaluating NTG 2.28: a positive value confirms NT, while a negative value indicates successful transfer. This combined evaluation ensures effective knowledge transfer while preventing performance degradation.

4.1.3 Algorithm

Algorithm 3 Offline Training

Input: Training dataset $\mathcal{S} = \{(x_s^i, y_s^i)\}_{i=1}^{n_s}$, Step sizes α, Number of epochs E **Output:** $\mathbf{w}_{SM}$: Optimal parameter for Source Model (SM)

1: Initialize $\mathbf{w}_{\text{SM}}$ randomly
2: **for** epoch = 1 to E **do**
3: Sample batch of points from $\mathcal{S}$
4: Evaluate $\nabla J_{\mathcal{S}}(\mathbf{w}_{\text{SM}})$ using $\mathcal{S}$
5: Update source model parameters with gradient descent:

$$\mathbf{w}_{SM} = \mathbf{w}_{SM} - \alpha \nabla J_{\mathcal{S}}(\mathbf{w}_{SM})$$

6: **end for**

With the number of required data points identified to compensate for errors and avoid negative transfer, we can now fine-tune the source model and achieve an adaptive model.

4.2 Experiments

This section describes the computational setup, datasets, and methodology used in this study. It begins with an overview of the hardware and software configurations, followed by a summary of the datasets employed for training, validation, and testing. Next, it covers source model selection, hyperparameter tuning, and comparative analysis. Exceptional buses are

Algorithm 4 Online Training

Input: $\mathbf{w}_{\text{SM}}, \gamma$, N_{TL}: Required number of data points for Transfer Learning (TL)
Output: $\mathbf{w}_{\text{AM}}$: Adapted parameters for new configuration

1: Initialize $\mathbf{w}_{\text{AM}} = \mathbf{w}_{\text{SM}}$: Use the source model's parameters as the initial knowledge
2: **while** system in operation **do**
3: Generate $\mathbf{N}_{\text{TL}}$ labeled target set $\mathcal{T}_{m,l} = \{(x^j_{m,l}, y^j_{m,l})\}_{j=1}^{N_{\text{TL}}}$ using Reverse Pass
4: Compute the adapted parameters with gradient descent:

$$\mathbf{w}_{\text{AM}} = \mathbf{w}_{\text{AM}} - \gamma \nabla J_{\mathcal{T}_{m,l}}(\mathbf{w}_{\text{AM}})$$

5: **end while**

then identified, and their cost variations are evaluated. The process of data generation for the second topology is outlined, ensuring alignment with optimality constraints. Additionally, strategies to mitigate negative transfer and determine the necessary number of data points for transfer learning are discussed. Finally, the section examines how transfer learning aids in adapting to topology changes.

4.2.1 Environment and Datasets

Experiments were conducted on a MacBook Pro with an Apple M2 processor, 16 GB RAM, using Python 3.9, TensorFlow 2.13.0, and Keras 2.13.1. The evaluation was performed on IEEE 14-bus, IEEE 118-bus, and PEGASE 1354-bus systems to assess scalability. The dataset was generated using MATPOWER's primal-dual interior point solver (MIPS) [62], with nodal active and reactive power demands randomly varied within 80% to 120% of nominal values, following standard practices [63, 64, 65].

For OPF, data was generated via MATPOWER's 'runopf' function with default cost function coefficients. Each system dataset contains 20,000 feasible scenarios, partitioned into 80% training, 10% validation, and 10% testing for balanced model evaluation [66].

4.2.2 Source Model

The source model is a fully connected feedforward neural network (FNN) that predicts bus voltages from simulated power system data. Its architecture consists of an input layer, a single hidden layer, and an output layer.

Model Architecture

The network processes power demand data to predict voltage magnitudes and angles. The input layer has n_{features} neurons corresponding to the feature vector, which includes power demand. The hidden layer, with a case-dependent number of units, uses the ReLU activation function:

$$\text{ReLU}(x) = \max(0, x)$$

The output layer, with n_{targets} neurons, produces continuous voltage predictions without an activation function.

Table 4.1: Model Parameters for Different Cases

Case	Input Layer Units	Hidden Layer Units	Output Layer Units	Total Parameters
14	28	256	27	14,363
118	236	256	235	121,067
1354	2708	1024	2707	5,547,228

All parameters are trainable, allowing the model to capture complex relationships between inputs and outputs effectively.

Preprocessing

To facilitate effective training, the data was scaled appropriately. Voltage angles were converted from degrees to radians, aligning their range with voltage magnitudes. Power demands were normalized by dividing by 100, corresponding to the system's base MVA. These transformations improved model convergence by ensuring consistent input scaling.

Additionally, branches causing network disconnection were removed to preserve system

integrity. The slack bus voltage angle was excluded from the output as it contained only zeros. Furthermore, branches connected to exceptional buses were filtered out, as the model could not be updated for them.

Loss Function and Optimization

The model is trained using the Mean Squared Error (MSE) loss function:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{\mathcal{N}} \sum_{i=1}^{\mathcal{N}} (y_i - \hat{y}_i)^2$$

where $\mathcal{N}$ is the number of samples, y_i is the true value, and $\hat{y}_i$ is the predicted value. Training is optimized using the Adam optimizer with a learning rate of 1×10^{-4}, selected for its adaptive learning rate and efficient handling of sparse gradients.

Training Procedure

The model is trained using 5-fold cross-validation for robust generalization. Each fold splits the data into training and validation sets, with training running for 200 epochs (batch size 20). Early stopping with a patience of 10 epochs prevents overfitting by halting training if validation loss stagnates.

To improve stability, multiple runs are performed per fold with different random seeds. The best model, based on validation loss, is saved after each fold for evaluation.

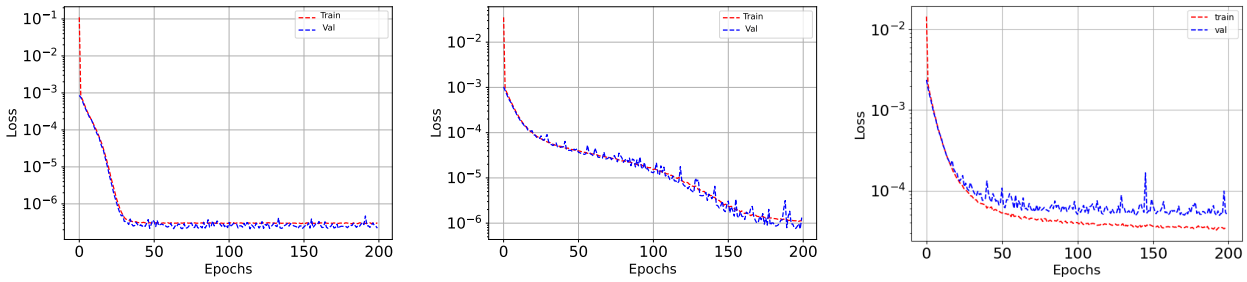
Results and Performance

The source model's performance is assessed using validation loss from each fold, with the final evaluation based on the average loss across all folds.

The neural network achieves comparable accuracy to classical methods while significantly reducing computation time, making it well-suited for real-time applications.

Table 4.2: Source Model Performance Metrics for Different Cases

Metric	Case 14	Case 118	Case 1354
Mean Validation MSE	2.38×10^{-7}	7.55×10^{-7}	1.13×10^{-4}
Best Validation Loss	2.10×10^{-7}	6.59×10^{-7}	2.97×10^{-5}
RMSE	0.00048	0.00081	0.00553
MAPE	0.08%	0.12%	2.69%
95% Confidence Interval	$(2.20 \times 10^{-7}, 2.56 \times 10^{-7})$	$(7.32 \times 10^{-7}, 7.79 \times 10^{-7})$	$(7.41 \times 10^{-5}, 1.52 \times 10^{-4})$
η^2	0.99	0.99	0.97
R^2 Score	0.99	0.99	0.99
MSE of Voltage Magnitude	1.0×10^{-8}	5.4×10^{-7}	3.59×10^{-5}
MSE of Voltage Angle	4.7×10^{-7}	7.7×10^{-7}	2.52×10^{-5}



(a) SM for IEEE 14-bus (b) SM for IEEE 118-Bus (c) SM for PEGASE 1354-bus

Figure 4.1: Training and validation loss over epochs for the optimal power flow prediction model. Both losses decrease steadily and converge, indicating good generalization and minimal overfitting. The loss is plotted on a logarithmic scale to emphasize the reduction in loss.

4.2.3 Adapting to New Grid Topology

To adapt to a new topology, the required amount of data is first determined while avoiding negative transfer. This is based on distributional differences and the source model's performance on new data. Data is then generated using reverse pass, followed by training the adapted model.

When a branch is removed from the power network without disconnecting any buses, the impact depends on whether the affected bus has a load. If the bus has a load, voltage and power generation remain fixed, demand is adjusted, and the model is fine-tuned. If the bus has no load, power injection equals generation, requiring no model adjustments—only cost variations are monitored.

The cost impact is evaluated using the economic dispatch objective function. In the IEEE 14-bus system, one bus has no demand, and removing one branch out of 19 removable

branches results in a 4.76% cost change. In the IEEE 118-bus system, eight generator buses have no associated demand, and removing 21 out of 179 generator-connected branches affects cost, while the rest have no impact. The average cost change is 0.32%, which is negligible.

Required Data for Transfer Learning: $N_{\mathbf{TL}}$

The required data points are determined iteratively by selecting a subset of branches and incrementally adding data until both the MSE and KLD fall below predefined thresholds, ensuring negative transfer does not occur. On average, only 8% of the data is needed.

Data Regeneration

The reverse pass generation process is applied to 8% of the data points, keeping bus voltages fixed while adjusting the power demands on the load and generation buses. The updated inputs are:

$$[v; \theta_{\mathcal{L},\mathcal{G}}] = f([P_d - \text{new}; Q_d - \text{new}])$$

This ensures consistency with system constraints while introducing variations.

Adapted Model

The Adapted Model (AM) shares the architecture of the pre-trained Source Model (SM) and is fine-tuned on the target dataset instead of being trained from scratch. It is trained for 15 epochs with a batch size of 5, with validation monitoring generalization performance. Leveraging transfer learning, AM benefits from prior knowledge, leading to faster convergence and improved accuracy, significantly reducing training time compared to the Target-Only Model (TOM).

Results and Performance

Table 4.3 compares the Adapted Model (AM) and the Target-Only Model (TOM) across three cases using key performance metrics. The results demonstrate the efficiency of transfer learning in reducing error and computational cost.

The AM achieves significantly lower Mean Squared Error (MSE) and Mean Absolute Percentage Error (MAPE) across all cases. In Case 14, its MSE is 4.00×10^{-7} compared to TOM's 7.08×10^{-3}, while MAPE remains below 0.5% for AM, far outperforming TOM. This trend persists in larger systems, with Case 1354 showing a substantial MAPE reduction from 34.43% (TOM) to 4.30% (AM), confirming the AM's superior accuracy.

The AM maintains a near-perfect R^2 score of 1.00 across all cases, while TOM slightly underperforms. The η^2 score further reinforces the effectiveness of transfer learning, particularly in Case 1354, where AM achieves 0.96 compared to TOM's 0.66.

A major advantage of AM is computational efficiency. Training times are significantly lower, with AM requiring only a few seconds compared to TOM's prolonged runtime (up to 81 minutes in Case 1354). This drastic reduction highlights the feasibility of transfer learning for real-time applications.

Overall, the results confirm that AM consistently outperforms TOM in accuracy, efficiency, and adaptability, making it a robust solution for handling topology changes in power grid prediction.

Table 4.3: Performance Metrics for AM and TOM Models Across Different Cases

Metric	**Case 14**		**Case 118**		**Case 1354**	
	AM	**TOM**	**AM**	**TOM**	**AM**	**TOM**
MSE	4.00×10^{-7}	7.08×10^{-3}	1.00×10^{-6}	7.23×10^{-4}	1.54×10^{-4}	2.60×10^{-3}
MAPE	0.13%	5.79%	0.14%	3.50%	4.30%	34.43%
R^2 **Score**	1.00	0.98	1.00	0.99	1.00	0.99
η^2	1.00	0.94	1.00	0.97	0.96	0.66
Training Time (s)	4.50 s	54 min	4.53 s	65 min	2.01 s	81 min

4.2.4 Comparison

Table 4.4 presents a comparative analysis of TL with existing Optimal Power Flow approaches. The comparison highlights the key limitations of state-of-the-art methods and demonstrates the advantages of TL in terms of model efficiency, generalization, and computational speed.

DeepOPF-FT: One Deep Neural Network for Multiple AC-OPF Problems With Flexible Topology

The **DeepOPF-FT** model [30] is designed to solve multiple AC-OPF problems using a deep neural network with flexible topology support. However, it has several drawbacks:

- **Requires branch features:** The model depends on line conductance g_{ij} and susceptance b_{ij}, reducing its flexibility.
- **More complex neural network:** The architecture is deep and computationally demanding.
- **Lower speed-up:** Achieves a limited **129**$\times$ (case 57) acceleration in OPF computation.
- **High data requirement:** For the IEEE 9-bus test system, which is relatively small, training requires a **random selection of 19,647 out of 32,768 possible topologies**, necessitating extensive datasets.

In contrast, TL provides significant improvements:

- **No branch feature dependency:** TL eliminates the need for g_{ij} or b_{ij}, enabling a simpler and more flexible model.
- **Simpler model:** Uses a single hidden layer, reducing computational complexity.
- **Higher speed-up:** Achieves superior acceleration:
  - **720**$\times$ (Case 14), reducing training time by **99.86%**.

  - **860.93**× (Case 118), reducing training time by **99.88%**.
  - **2417.91**× (Case 1354), reducing training time by **99.96%**.

- **Lower data requirement:** TL utilizes only **20,000 training samples for the source model and 1,600 for the adapted model**, significantly reducing dataset dependency.

Meta-Learning Approach to OPF Under Topology Reconfigurations

The **Meta-Learning Approach** to OPF proposed in [38] focuses on adapting to different network topologies. However, it suffers from the following limitations:

- **High topology dependency:** Requires **70 topology variations** for source training, making adaptation effort-intensive.
- **More complex model:** Needs a deep network and extensive training data.
- **Higher MSE:** Achieves an MSE of **0.003**, which is significantly higher than TL.
- **Cost-sensitive approach:** Accounts for cost variations, whereas in our data generation approach, the cost remains unchanged.
- **Lower generalization:** Achieves an η^2 score of **0.97**, indicating reduced adaptability.

On the other hand, TL offers substantial benefits:

- **Single topology requirement:** Used an unchanged topology for the source model.
- **Simpler model:** Uses a single hidden layer while maintaining accuracy.
- **Lower MSE:** Achieves an **MSE of** 1.00×10^{-6}, vastly improving precision.
- **Stable cost behavior:** In the reverse pass generation process, the cost remains unchanged. .

- **Better generalization:** Achieves an η^2 score of **0.99**, indicating improved model reliability.

Topology-Aware Graph Neural Networks for Learning Feasible and Adaptive AC-OPF Solutions

The **Topology-Aware Graph Neural Network (GNN)** [8] employs a graph-based learning framework to solve OPF. However, it has several limitations:

- **Higher MSE:**
  - **Case 118:** 3.4×10^{-3}.
  - **Case 1354:** 5.5×10^{-3}.
- **Higher complexity:** Requires multiple components, including graph convolutions, topology-based filters, and message-passing mechanisms.
- **High data requirement:** Requires substantial labeled datasets for effective training.
- **Difficult deployment:** Relies on specialized graph-based ML frameworks, making real-world implementation challenging.

TL addresses these challenges through the following improvements:

- **Lower MSE:** TL achieves significantly better accuracy:
  - **Case 118:** 1.00×10^{-6}.
  - **Case 1354:** 1.54×10^{-4}.
- **Simplified model:** TL is a **fully connected neural network (FCNN)**, eliminating complex GNN processing.
- **Lower data dependency:** TL requires fewer labeled samples, making it more practical for real-world OPF applications.

- **Easier deployment:** TL integrates seamlessly into **existing ML workflows**, without the need for graph-based learning tools.

Summary

From the comparative analysis, it is evident that TL surpasses existing OPF approaches in several key aspects. TL eliminates reliance on branch features, reduces neural network complexity, and significantly reduces the training dataset requirements. Additionally, TL achieves significantly higher speed-up factors while maintaining superior accuracy with lower MSE. Furthermore, its ease of integration into existing machine learning pipelines makes it a more practical solution compared to graph-based and meta-learning approaches.

Thus, **Transfer Learning presents a more computationally efficient, data-efficient, and scalable alternative** to state-of-the-art OPF methodologies.

Table 4.4: Comparison of TL with Existing OPF Approaches

Approach	Limitations	Advantages of Transfer Learning (TL)
DeepOPF-FT (Low et al., 2023) [30]	• **Requires branch features** • **Complex model:** Deep NN with high computation. • **Lower speed-up:** Only **129×** (case 57) acceleration. • **High data requirement**	• **No branch feature dependency** • **Simpler model:** Single hidden layer, lower computation. • **Higher speed-up:** **720×** (Case 14), **860.93×** (Case 118), **2417.91×** (Case 1354). • **Lower data requirement**
Meta-Learning OPF (Chen et al., 2022) [38]	• **High topology dependency** • **Complex model:** Deep network, large dataset. • **Higher MSE:** 3×10^{-3}, significantly higher than TL. • **Cost-sensitive** • **Lower generalization:** η^2 score of **0.97**.	• **Single topology requirement** • **Simpler model:** Single hidden layer, reduced complexity. • **Lower MSE:** 1×10^{-6}, significantly improved. • **Stable cost behavior** • **Better generalization:** η^2 score of **0.99**.
Topology-Aware GNN for OPF (Liu et al., 2022) [8]	• **Higher MSE:** – **Case 118:** 3.4×10^{-3}. – **Case 1354:** 5.5×10^{-3}. • **High complexity:** Requires graph convolutions, filters, message passing. • **High data requirement:** Needs extensive labeled datasets. • **Difficult deployment:** Requires specialized ML frameworks.	• **Lower MSE:** – **Case 118:** 1.00×10^{-6}. – **Case 1354:** 1.54×10^{-4}. • **Simpler model:** Fully connected NN, no graph processing. • **Lower data dependency:** Requires fewer labeled samples. • **Easier deployment:** No specialized ML frameworks needed.

Chapter 5

Conclusion

5.1 Summary and Contributions

In this thesis, we proposed a transfer learning-based framework to enhance power flow and optimal power management in grids with substantial renewable energy integration. Our approach enables efficient adaptation to changing grid topologies through a novel data regeneration method, significantly reducing the need for extensive retraining. Experimental evaluations demonstrated notable improvements, achieving accurate adaptation with only 6% of the data while maintaining computation times under 3 seconds. These results indicate the potential of our approach for real-world applications, contributing to improved grid reliability and operational efficiency.

Our method can replace computationally expensive processes involved in power system planning, security assessment, grid management, and optimization. It can be integrated into existing solvers or used to develop new tools for utility companies, grid operators, and urban planners. The approach is highly scalable and fast, making it suitable for large-scale deployments. Furthermore, it supports the integration of sustainable energy solutions, directly aligning with Ontario's, Canada's, and global sustainability goals. For example, collaboration with institutions like York University in the York Region, which serves a 50,000-member

community, demonstrates how data-driven methods can enable sustainability and smarter energy planning.

5.2 Future Work

Future research could explore the integration of unsupervised learning techniques to further reduce dependency on labeled data. Additionally, incorporating Physics-Informed Learning could enhance model generalization by embedding physical constraints directly into the learning process. However, even physics-guided models must account for system topology; ignoring it would result in incomplete representations of the physical system, as topology defines the connectivity and power flow between nodes. If the topology changes but is not reflected in the model, the embedded physical constraints may no longer hold, limiting the model's reliability. Another promising direction is the extension of transfer learning to other power system challenges, such as stability assessment, fault detection, and demand forecasting.

Furthermore, ensuring the robustness of ML-based grid management systems against adversarial attacks, cyber threats, and data integrity issues is crucial. Developing anomaly detection methods and fault-tolerant learning mechanisms could significantly enhance system security and resilience. These advancements would further improve the scalability and applicability of ML-driven grid management solutions, enabling more adaptive and intelligent power systems.

Overall, this field of research remains rich with potential, offering numerous unexplored opportunities for future investigation.

Bibliography

[1] Global Affairs Canada. *Canada's International Development Agenda*. https://www.international.gc.ca/world-monde/issues_development-enjeux_developpement/priorities-priorites/agenda-programme.aspx?lang=eng. [Accessed: Aug. 20, 2024].

[2] Government of Canada. *Canada's climate plan: A healthy environment and a healthy economy*. https://www.canada.ca/en/services/environment/weather/climatechange/climate-plan/net-zero-emissions-2050.html. [Accessed: Aug. 20, 2024].

[3] Government of Canada. *National Inventory Report 1990–2021: Greenhouse Gas Sources and Sinks in Canada*. https://www.canada.ca/en/environment-climate-change/services/climate-change/greenhouse-gas-emissions/inventory.html. Environment and Climate Change Canada. 2023.

[4] Stéphane Fliscounakis et al. "Contingency ranking with respect to overloads in very large power systems taking into account uncertainty, preventive, and corrective actions". In: *IEEE Transactions on Power Systems* 28.4 (2013), pp. 4909–4917.

[5] Hadi Saadat et al. *Power system analysis*. Vol. 2. McGraw-hill, 1999.

[6] Brian Stott. "Review of load-flow calculation methods". In: *Proceedings of the IEEE* 62.7 (1974), pp. 916–929.

[7] Soobae Kim and Thomas J Overbye. "Hybrid power flow analysis: Combination of AC and DC models". In: *2011 IEEE Power and Energy Conference at Illinois*. IEEE. 2011, pp. 1–4.

[8] Shaohui Liu, Chengyang Wu, and Hao Zhu. "Topology-aware graph neural networks for learning feasible and adaptive AC-OPF solutions". In: *IEEE Transactions on Power Systems* 38.6 (2022), pp. 5660–5670.

[9] Junfei Wang and Pirathayini Srikantha. "Fast optimal power flow with guarantees via an unsupervised generative model". In: *IEEE Transactions on Power Systems* 38.5 (2022), pp. 4593–4604.

[10] Mary B Cain, Richard P O'neill, Anya Castillo, et al. "History of optimal power flow and formulations". In: *Federal Energy Regulatory Commission* 1 (2012), pp. 1–36.

[11] Carl T Kelley. *Solving nonlinear equations with Newton's method*. SIAM, 2003.

[12] John J Grainger. *Power system analysis*. McGraw-Hill, 1999.

[13] Thomas J Overbye, Xu Cheng, and Yan Sun. "A comparison of the AC and DC power flow models for LMP calculations". In: *37th Annual Hawaii international conference on system sciences, 2004. Proceedings of the*. IEEE. 2004, 9–pp.

[14] William F Tinney and Clifford E Hart. "Power flow solution by Newton's method". In: *IEEE Transactions on Power Apparatus and systems* 11 (1967), pp. 1449–1460.

[15] Geraldo Leite Torres and Victor Hugo Quintana. "An interior-point method for nonlinear optimal power flow using voltage rectangular coordinates". In: *IEEE transactions on Power Systems* 13.4 (1998), pp. 1211–1218.

[16] Florin Capitanescu et al. "Interior-point based algorithms for the solution of optimal power flow problems". In: *Electric Power systems research* 77.5-6 (2007), pp. 508–517.

[17] Luciana Casacio, Christiano Lyra, and Aurelio RL Oliveira. "Interior point methods for power flow optimization with security constraints". In: *International Transactions in Operational Research* 26.1 (2019), pp. 364–378.

[18] James A Momoh and JZ Zhu. "Improved interior point method for OPF problems". In: *IEEE Transactions on Power Systems* 14.3 (1999), pp. 1114–1120.

[19] Qian Gao et al. “Model-driven architecture of extreme learning machine to extract power flow features”. In: *IEEE Transactions on Neural Networks and Learning Systems* 32.10 (2020), pp. 4680–4690.

[20] Wenqian Dong et al. “Smart-PGSim: Using neural network to accelerate AC-OPF power grid simulation”. In: *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE. 2020, pp. 1–15.

[21] Thuan Pham and Xingpeng Li. “Neural network-based power flow model”. In: *2022 IEEE Green Technologies Conference (GreenTech)*. IEEE. 2022, pp. 105–109.

[22] Xinyue Hu et al. “Physics-guided deep neural networks for power flow analysis”. In: *IEEE Transactions on Power Systems* 36.3 (2020), pp. 2082–2092.

[23] Fouad Hasan, Amin Kargarian, and Ali Mohammadi. “A survey on applications of machine learning for optimal power flow”. In: *2020 IEEE Texas Power and Energy Conference (TPEC)*. IEEE. 2020, pp. 1–6.

[24] 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.

[25] Kejun Chen, Shourya Bose, and Yu Zhang. “Unsupervised deep learning for AC optimal power flow via Lagrangian duality”. In: *GLOBECOM 2022-2022 IEEE Global Communications Conference*. IEEE. 2022, pp. 5305–5310.

[26] Wanjun Huang, Minghua Chen, and Steven H Low. “Unsupervised Learning for Solving AC Optimal Power Flows: Design, Analysis, and Experiment”. In: *IEEE Transactions on Power Systems* (2024).

[27] Damian Owerko, Fernando Gama, and Alejandro Ribeiro. “Unsupervised optimal power flow using graph neural networks”. In: *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2024, pp. 6885–6889.

[28] Akshay S Korad and Kory W Hedman. "Robust corrective topology control for system reliability". In: *IEEE Transactions on Power Systems* 28.4 (2013), pp. 4042–4051.

[29] Ming Dong, Daniel Wiebe, and Jian Shi. "An accelerated and risk-free AC power flow method with machine learning based initiation". In: *2022 IEEE Electrical Power and Energy Conference (EPEC)*. IEEE. 2022, pp. 103–108.

[30] Min Zhou, Minghua Chen, and Steven H Low. "DeepOPF-FT: One deep neural network for multiple AC-OPF problems with flexible topology". In: *IEEE Transactions on Power Systems* 38.1 (2022), pp. 964–967.

[31] Tania B Lopez-Garcia and José Antonio Domínguez-Navarro. "Graph Neural Network Power Flow Solver for Dynamical Electrical Networks". In: *2022 IEEE 21st Mediterranean Electrotechnical Conference (MELECON)*. IEEE. 2022, pp. 825–830.

[32] Tania B Lopez-Garcia and José A Domínguez-Navarro. "Power flow analysis via typed graph neural networks". In: *Engineering Applications of Artificial Intelligence* 117 (2023), p. 105567.

[33] Fuzhen Zhuang et al. "A comprehensive survey on transfer learning". In: *Proceedings of the IEEE* 109.1 (2020), pp. 43–76.

[34] Thijs Peirelinck et al. "Transfer learning in demand response: A review of algorithms for data-efficient modelling and control". In: *Energy and AI* 7 (2022), p. 100126.

[35] Haoran Li, Zhihao Ma, and Yang Weng. "A transfer learning framework for power system event identification". In: *IEEE Transactions on Power Systems* 37.6 (2022), pp. 4424–4435.

[36] Chao Ren and Yan Xu. "Transfer learning-based power system online dynamic security assessment: Using one model to assess many unlearned faults". In: *IEEE Transactions on Power Systems* 35.1 (2019), pp. 821–824.

[37] Chao Ren et al. "An integrated transfer learning method for power system dynamic security assessment of unlearned faults with missing data". In: *IEEE Transactions on Power Systems* 36.5 (2021), pp. 4856–4859.

[38] Yexiang Chen et al. "A meta-learning approach to the optimal power flow problem under topology reconfigurations". In: *IEEE Open Access Journal of Power and Energy* 9 (2022), pp. 109–120.

[39] Michael T Rosenstein et al. "To transfer or not to transfer". In: *NIPS 2005 workshop on transfer learning*. Vol. 898. 3. 2005.

[40] Math HJ Bollen and Fainan Hassan. *Integration of distributed generation in the power system*. John wiley & sons, 2011.

[41] J Duncan Glover, Thomas J Overbye, and Mulukutla S Sarma. *Power system analysis & design*. Cengage Learning, 2017.

[42] Richard E Brown. *Electric power distribution reliability*. CRC press, 2017.

[43] Allen J Wood, Bruce F Wollenberg, and Gerald B Sheblé. *Power generation, operation, and control*. John wiley & sons, 2013.

[44] Prabha Kundur. "Power system stability". In: *Power system stability and control* 10.1 (2007), pp. 7–1.

[45] Illinois Center for a Smarter Electric Grid (ICSEG). *IEEE 14-Bus System*. `https://icseg.iti.illinois.edu/ieee-14-bus-system/`. Accessed: 2024-09-03. 2013.

[46] J Carpentier. "Contribution to the economic dispatch problem". In: *Bulletin de la Societe Francoise des Electriciens* 3.8 (1962), pp. 431–447.

[47] Vijay Vittal and Arthur R Bergen. "Power systems analysis". In: *Prentice Hall* (1999), pp. 1–2.

[48] Javad Lavaei and Steven H Low. "Zero duality gap in optimal power flow problem". In: *IEEE Transactions on Power systems* 27.1 (2011), pp. 92–107.

[49] Stephen Frank, Ingrida Steponavice, and Steffen Rebennack. "Optimal power flow: A bibliographic survey I: Formulations and deterministic methods". In: *Energy systems* 3 (2012), pp. 221–258.

[50] Stephen Frank, Ingrida Steponavice, and Steffen Rebennack. "Optimal power flow: A bibliographic survey I: Formulations and deterministic methods". In: *Energy systems* 3 (2012), pp. 221–258.

[51] Andreas Venzke et al. "Learning optimal power flow: Worst-case guarantees for neural networks". In: *2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. IEEE. 2020, pp. 1–7.

[52] Santisudha Panigrahi, Anuja Nanda, and Tripti Swarnkar. "A survey on transfer learning". In: *Intelligent and Cloud Computing: Proceedings of ICICC 2019, Volume 1*. Springer. 2021, pp. 781–789.

[53] Zirui Wang et al. "Characterizing and avoiding negative transfer". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 11293–11302.

[54] Wen Zhang et al. "A survey on negative transfer". In: *IEEE/CAA Journal of Automatica Sinica* 10.2 (2022), pp. 305–329.

[55] Maxime Oquab et al. "Learning and transferring mid-level image representations using convolutional neural networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 1717–1724.

[56] Jason Yosinski et al. "How transferable are features in deep neural networks?" In: *Advances in neural information processing systems* 27 (2014).

[57] Jeremy Howard and Sebastian Ruder. "Universal language model fine-tuning for text classification". In: *arXiv preprint arXiv:1801.06146* (2018).

[58] Jeff Donahue et al. “Decaf: A deep convolutional activation feature for generic visual recognition”. In: *International conference on machine learning*. PMLR. 2014, pp. 647–655.

[59] Solomon Kullback and Richard A Leibler. “On information and sufficiency”. In: *The annals of mathematical statistics* 22.1 (1951), pp. 79–86.

[60] Boqing Gong et al. “Geodesic flow kernel for unsupervised domain adaptation”. In: *2012 IEEE conference on computer vision and pattern recognition*. IEEE. 2012, pp. 2066–2073.

[61] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*. Vol. 4. 4. Springer, 2006.

[62] Ray Daniel Zimmerman, Carlos Edmundo Murillo-Sánchez, and Robert John 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.

[63] Ferdinando Fioretto, Terrence WK Mak, and Pascal Van Hentenryck. “Predicting ac optimal power flows: Combining deep learning and lagrangian dual methods”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 34. 01. 2020, pp. 630–637.

[64] Xiang Pan et al. “DeepOPF: A feasibility-optimized deep neural network approach for AC optimal power flow problems”. In: *IEEE Systems Journal* 17.1 (2022), pp. 673–683.

[65] Wanjun Huang et al. “Deepopf-v: Solving ac-opf problems efficiently”. In: *IEEE Transactions on Power Systems* 37.1 (2021), pp. 800–803.

[66] Pirathayini Srikantha. *Power Flow Dataset*. `http://www.cse.yorku.ca/~psrikan/pf_dataset.html`. Accessed: 30 September 2024.