PSRC Working Group C43 Report

October 2023

PRACTICAL APPLICATIONS OF ARTIFICIAL INTELLIGENCE / MACHINE LEARNING IN POWER SYSTEM PROTECTION AND CONTROL

Chair: Yi Hu
Vice Chair: Adi Mulawarman
Secretary: Zheyuan Cheng

Members and Contributors

Abder Elandaloussi
Alex Apostolov
Ali Bidram
Athula Rajapakse
Carolina Arbona
Dan Sabin
Jayaprakash Ponraj
Jean Raymond
Jörg Blumschein
Juan F. Piñeros S.
Matthew Reno
Nirmal Nair
Ratan Das
Robert Fowler
Sebastien Billaut
Sukumar Brahma
Sukumar Kamalasadan
Vahid Madani
Yu Liu
Yujie Yin

KEYWORDS

Artificial Intelligence (AI), Control, Machine Learning (ML), Power System, Practical Application, Protection, Relaying

CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE LEFT BLANK INTENTIONALLY

1 INTRODUCTION

In practical terms, intelligence can be defined in many ways: advanced comprehension, a capacity to further the existing reasoning, demonstrative knowledge, and learned decision-making. Intelligence can be demonstrated using machines, similar to the natural intelligence shown by humans. Adaptive development is required at various stages, such as cognition, manipulation, rationalization, communication, and reaction to any common transaction. Here, the continuous learning experience facilitates the challenge of automated enhancement in overall system performance over time.

Artificial intelligence (AI) is the simulation of human intelligence in machines that think and act like humans. An effective AI application requires many skill sets such as cognition, manipulation, rationalization, communication, and reaction to be incorporated into the scheme.

AI utilizes Machine Learning (ML) and other associated techniques such as Heuristics to resolve real challenges. Various computational tools used for implementing these skill sets include search and optimization, artificial neural network, fuzzy logic, probabilistic methods for uncertain reasoning, reinforcement learning, and other supervised and unsupervised learning methods. ML is a subfield of artificial intelligence that enables the gathering and analyzing volumes of data to extract representative features based on appropriate training (learning) and develop an equation or algorithm for deriving useful information or action. As an example, text recognition within images is now a commonly used tool (OCR, optical character recognition) that is powered by ML. OCR models can search for the pertinent information using representative features to identify text predictively rather than programmatically.

ML or its earlier vintages of 'Artificial Neural Network' has been in practice within the power system industry, especially in operations, workforce management, and planning over a considerable period of time. Some examples are forecasting, optimization, scheduling, and unit commitment. Due to the recent digital transformation progression in the areas such as Cloud Computing (CC), Internet of Things (IoT), and the use of ML to expand the earlier techniques, an upsurge of AI or ML (sometimes used interchangeably, also in this report) is conveying additional significance in many power system applications.

This report aims to introduce significant AI/ML technical concepts to a Protection and Control (P&C) knowledge base and audience. Providing the foundation for a methodical approach to incorporate this new technology within our industry while displaying representative examples of possible applications.

This report initiates the reader on current developments and futuristic ideas in AI/ML, to spark creativity in advancing P&C applications within Power System Industry. For example:

- AI/ML can help detect and locate high impedance faults faster.

- AI/ML can assist in predicting and isolating potential component failures by utilizing partial discharge sensor data.
- AI/ML can be used to evaluate challenges to the dynamic stabilities of the power network and support decision-making by using Phasor Measurement and Control Unit data and architecture, thus mitigating the stability control challenges due to increasing low inertia resources.

The following section describes the report's detailed purpose, scope, and limitations as the topic of artificial intelligence in the power system are very broad.

2 SCOPE

This report covers the practical applications of AI/ML in the protection and control of power systems. The process of applying AI/ML for solving protection or control challenges in the power system involves the design, development, validation, integration, field testing, and deployment of AI/ML models. All these applications will focus on reliability, availability, dependability, security, speed, and accuracy to support the operation of a dynamic power system. The report will primarily provide insights from data collection to detailed analytics and informed decision-making to support asset and network protection and anticipate potential disturbances to lessen the impacts. The protection and control applications in this report will be limited to the protection of the electric network or the power assets.

The focus will be AI/ML applications in P&C with updates on:

- Basics of the AI/ML technology
- Practical applications in use and Emerging application areas
- Challenges of applying the technology in power system protection and control
- Considerations in developing, validating, field testing, and implementing practical applications
- Risks, challenges and acceptance criteria for the use of AI/ML for protection

The report is not intended to be used as a technical guide but as a resource that provides recent advances in AI/ML applications in the P&C area of power system.

3 AI/ML Technology

Since the beginning of computing technology, the idea of imitating human capabilities has been a goal of computational development. In 1950, Alan Turing created a machine to do a complex task faster than a human being, and established the principles of machine intelligence [72]. There have been several definitions of artificial intelligence through its evolution. Any human creation or effort to imitate a human skill in executing a complex task in a wide range of environments to achieve a specified objective is considered artificial intelligence [74].

To perform complex tasks, several types of algorithms have been developed. Based on the type of human skills that these algorithms or techniques imitate; they can be grouped as:

- Reasoning through bodies of knowledge – imitates the human's ability to apply prior knowledge in similar situations via inference and reasoning.
- Learning by experience - imitates the human's ability to learn by experience to perform pattern recognition, classification, regression tasks, etc.
- Learning by evolution – imitates human's ability to learn by trial and error by considering actions and an objective that is tested for success.

The role of AI/ML technology in a complex system can vary depending on the nature of the application, and some of the common elementary functions performed by the technology include:

- Pattern recognition: application of one or more techniques or algorithms to find, based on some criteria, a similar 'pattern' or 'class type' in a set of data.
- Classification: identification of the 'class type' of given data, according to a predefined set of class types.
- Clustering: grouping of data based on the similarity of their characteristics
- Regression and function approximation: a statistical or learning process to create a functional relationship between two or more correlated variables from data, often to predict values of one variable when given values of the others.

3.1 Reasoning through bodies of knowledge

Reasoning based on knowledge is a key skill of human intelligence. Starting with conventional logic based on if-then clauses, algorithms and techniques have been developed to imitate this skill. These systems can capture the scarce expertise and/or difficult-to-use sources of knowledge to perform specific tasks in a much faster and more consistent way than a human user. The main approaches to using knowledge-based reasoning are expert systems, case-based reasoning, and Fuzzy logic.

Expert Systems

An expert system consists of a knowledge base, an inference engine, and a user interface, as shown in Figure 3-1. The process of extracting knowledge from the subject experts and constructing a knowledge base is known as knowledge engineering. The knowledge base stores both factual knowledge (widely accepted information) and heuristic knowledge (information derived from an expert's judgment, practice, and ability to guess and evaluate) in the form of a set of if-then rules such as:

R1: IF (I > 2 pu) THEN (I is High) R2: IF (V < 0.9 pu) THEN (V is Low)
R3: IF (I is High) THEN (Fault = True) R4: IF (V is Low) THEN (Fault = True)

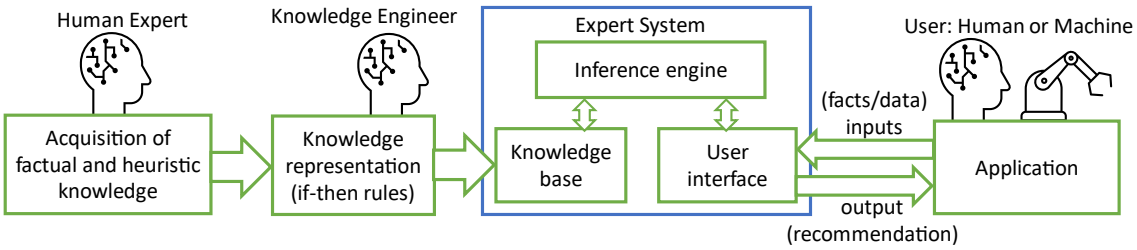


Figure 3-1: Basic elements of an expert system

The inference engine acquires and manipulates the knowledge from the knowledge base to arrive at a particular solution. This is achieved by applying rules repeatedly to the presented facts (inputs) and resolving conflicts when multiple rules are applicable to a particular case. The inference engine may add new knowledge to the knowledge base if required. Depending on the nature of the problem, inference engines use either forward chaining or backward chaining of rules to recommend a solution.

In forward chaining, the inference engine attempts to conclude based on given facts; rules are applied to derive new facts as conclusions (forward chaining of rules) until the requested final goal fact is found. It is data-driven and commonly used to solve more open-ended problems such as prediction, control, design, or planning.

In backward chaining, the inference engine attempts to find the conditions (facts) that lead to a given conclusion; it strives to match the assumed conclusion (the goal or subgoal) with the conclusion (then part) of the rules, and if such a rule is found, its premise (if part) becomes the new subgoal. The process is continued until the goal or subgoal is grounded in initial facts. Backward chaining is best suited for applications in which the possible conclusions are limited in number and well defined. Examples are classification or diagnosis type systems.

Case-based reasoning

In case-based reasoning, the knowledge is represented using real cases, and when faced with new cases or conditions, decisions are made based on the solutions to similar past problems. Case-based reasoning is accomplished by gathering case histories and is implemented by identifying significant features that describe a case. Case-based reasoning systems offer the capability of incremental, sustained learning; each time a problem is solved, a new experience is retained and can be applied for future problems. In general, the case-based reasoning process entails:

1. Retrieve: Gathering from memory an experience closest to the current problem.
2. Reuse: Suggesting a solution based on the experience and adapting it to meet the demands of the new situation.
3. Revise: Evaluating the use of the solution in the new context.
4. Retain: Storing this new problem-solving method in the memory system

Fuzzy logic systems

Conventional expert systems are based on Boolean logic where only two possibilities are considered; a rule can be either true or false under a given condition. In the real world, problems are often not precisely defined, and the data can be uncertain or ambiguous. Thus, it is hard to establish whether a certain statement (or rule) is True or False. Fuzzy logic, originally invented by Lotfi Zadeh in 1965, combines numerical data and linguistic knowledge to offer the flexibility of reasoning, especially when encountered with uncertainty [76].

The principle is to work with degrees of possibilities instead of only two possibilities considered in the conventional logic as a comparative approach to the real-life environment [76]. For example, a voltage of 0.9 pu can be considered 'Normal' to a degree of 70%, and at the same time, it can be considered 'Low' to a degree of 30%.

A fuzzy system consists of a rule base, an inference engine, a fuzzifier, and a defuzzifier, as shown in Figure 3-2 (a). The process of assigning linguistic labels and membership values for a crisp variable such as a voltage is called fuzzification. It is achieved using a set of membership functions, each denoted by a linguistic label, as shown in Figure 3-2 (b). Defuzzification is the opposite process. The rule base consists of if-then rules written in terms of the linguistic labels of the inputs and outputs, for instance:

IF (V is Normal) THEN (Stability Margin is High)
IF (V is Low) THEN (Stability Margin is Very Low)

The inference engine uses a rule base to evaluate the membership values of the output variable(s) for the given inputs. Then the process of defuzzification is performed if a crisp output value is desired.

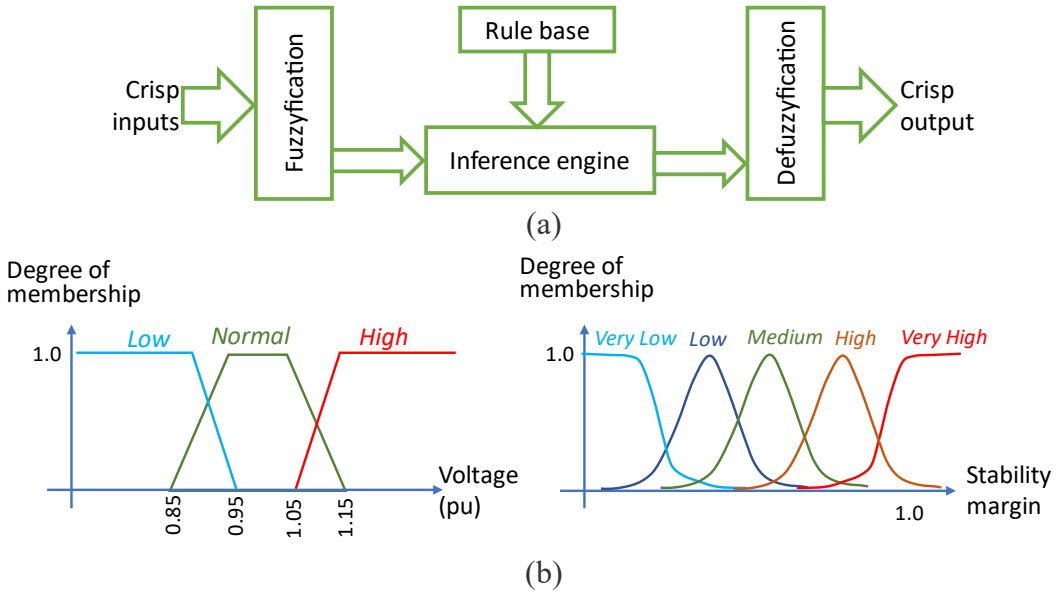


Figure 3-2: (a) Components of a fuzzy inference system (b) Examples of membership functions

3.2 Learning by experience

The techniques and algorithms that accomplish learning by experience are broadly considered as machine learning methods. In this type of learning, machine learning techniques or algorithms process input data in order to perform classification, pattern recognition, clustering, regression tasks, and more, based on the experience learned from the data [76].

Artificial Neural Networks (ANN) are at the forefront of machine learning as enablers to learning by experience. Warren McCulloch and Walter Pitts laid the foundations for the artificial neural networks in 1943 by developing the first mathematical model inspired by biological neurons and by Frank Rosenblatt in 1957 by creating the perceptron model as an effort to imitate the human brain. A commonly used model of an artificial neuron is shown in Figure 3-3 (a). An artificial neural network is a network of neurons consisting of an input layer, which receives data from outside sources (data files, sensors, etc.), one or more hidden layers that process the data, and an output layer that provides one or more output data points based on the function of the network. An example of a feedforward neural network is shown in Figure 3-3 (b). The weights of the neurons are adjusted considering the relationship between the inputs and the outputs as defined by the experience data [73].

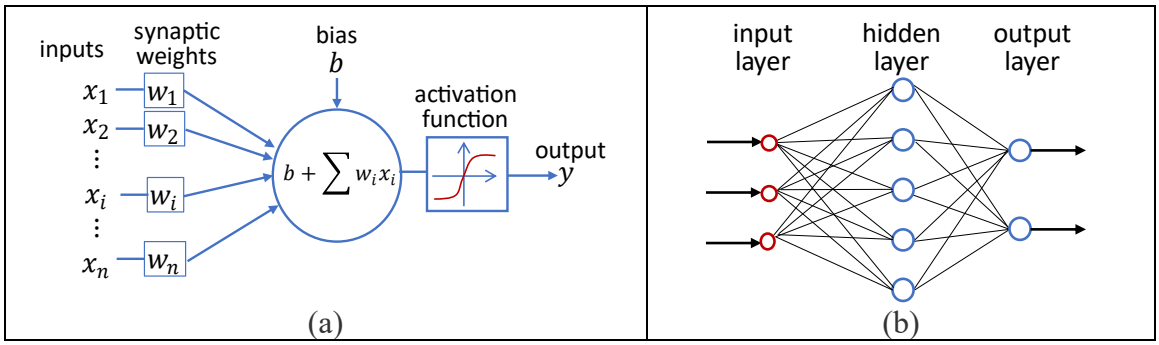


Figure 3-3: (a) Model of an artificial neuron (b) Example of a feedforward neural network

There are several kinds of ANN architectures; among them, the main configurations are:

- Single Layer Feedforward Architecture
- Multiple Layer Feedforward Architecture
- Recurrent or Feedback Architecture
- Meshed Architecture

In addition to ANNs, other structures such as decision trees, support vector machine, etc., can facilitate learning by experience.

After defining the structure or architecture of an ANN (the number of hidden layers, the number of neurons in each layer, etc.), it needs to be trained. Training is the process of incorporating the experience (gathered from the training data) into trainable parameters of

the ANN. There are several approaches for training ANNs. They can be classified based on the approach used. The main types of training approaches include:

- Supervised Learning – data used for training is tagged by a human or available as pairs of inputs and the corresponding expected outputs. Typically used for classification and regression applications.
- Unsupervised Learning – self-organization to capture patterns in input data as neuronal predilections or probability densities. Three main domains of applications are clustering, association rules learning and dimensionality reduction.
- Reinforcement Learning – in model-free RL, the machine learns by *interacting* with the environment as an agent and is given a numerical performance score (reward) after performing the action as its guidance. Model-free RL is popularly used in robotics. Model-based reinforcement learning algorithms can *model* the environment to predict the outcomes of actions before performing them and optimize iteratively from these predictions. Model-free RL can be used in situations where the environment is better known, such as playing chess.

The availability of the input data used to train the algorithm will be different for each application. Sometimes, the entire dataset is available from the beginning; however, the data that the algorithms need will grow over time. As a result, the learning by experience approach taken considers the data availability and is broken down into offline and online learning. When there is not enough data (experience) to perform an offline learning, the focus normally is about recognition of types or classes (for example clustering) by learning online with the input data.

- Offline Learning – batch learning techniques which generate the best ANN parameters by learning on the entire training data set at once
- Online Learning – sequential learning in which data becomes available in a sequential order and is used to update the ANN weights at each step. Online learning is used where it is necessary for the algorithm to dynamically adapt to new patterns in the data (distribution shift) or when it is infeasible to learn from an entire dataset.

Normally, a neural network architecture will be chosen based on the application and data. The types, size, and number of layers within the network, along with the optimization strategy and loss equation being carefully chosen to allow the network to efficiently map the input data to the application goal.

Figure 3-4 is an overview of types of machine learning algorithms, and Figure 3-5 is the Scikit-learn algorithm cheat sheet [77].

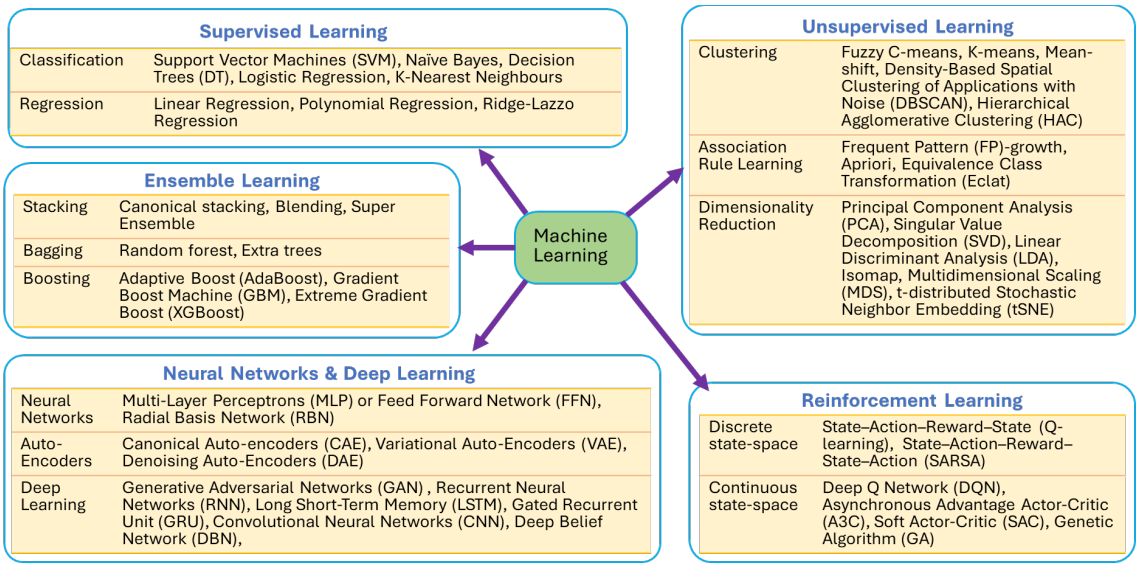


Figure 3-4: Types of Machine learning algorithms

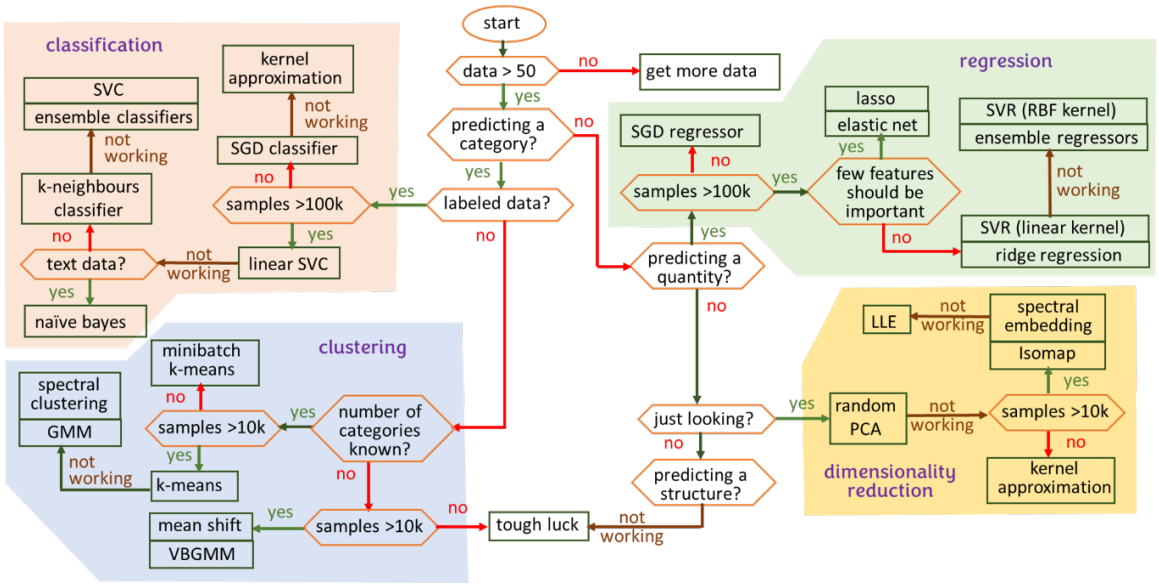


Figure 3-5: Scikit-learn Algorithm Cheat Sheet [112]

3.3 Learning by evolution

Trial and error approaches have been a key part to human learning. Several techniques have been developed to imitate this human skill and focus on an objective while using the experience to find a solution for a problem, by means of the evolution of a population.

The population in the context of evolutionary learning refers to a set of potential solutions to a problem. For example, for the problem of finding setting of a non-directional overcurrent relay (ANSI 51), a solution could be a pair of values representing the pickup current and the time dial. The objective for application of device 51 to a transformer could

be to get a reference operating time for the internal and external faults. Given a range of short circuit currents, the operating time function can be evaluated for potential solutions for this individual in a population. A learning by evolution algorithm applies random changes guided by natural evolutionary principles, swarm intelligence, colony behavior or physical/chemical processes to find a group of solutions (in this case settings) to meet the objective (the required operating times) or the best possible solution. Starting with an initial population (often randomly generated), these kinds of algorithms can find better solutions for complex problems by making variations in every population.

Evolutionary computing started in 1960s with the evolutionary strategies proposed by Rechenberg, evolutionary programming proposed by Fogel, and the idea of the genetic algorithms, which is a population-based algorithm, proposed by Holland [76]. In general, the ideas of evolution in computing are based on the processes observed in biological environments and focus mainly on cellular behaviors and multiple groups – colony behaviors [74].

Metaheuristic algorithms are a set of techniques or algorithms applied to solve optimization and search problems using population evolution. Some of the popular algorithms are Simulated Annealing, Ants Colony, Bees Colony, Genetic Algorithms, Differential Evolution, Particle Swarm Optimization, Harmony Search, Bat Algorithm, and Taboo Search. Metaheuristics are widely used to solve complex optimization problems with a single objective or multi-objective focus and the capability of problem generalization.

3.4 Deep Learning

Deep learning is a subfield of machine learning and its backbone is built on neural networks. In fact, it is the number of node layers, or depth, of neural networks that distinguishes a shallow neural network from a deep learning algorithm. A deep neural network will have more than three layers. These algorithms use multiple layers to progressively extract higher-level features from the raw input, with minimal human intervention. Learning can be supervised, semi-supervised or unsupervised.

Traditional machine learning algorithms such as linear regression, logistic regression, shallow neural networks, and decision trees are algorithms that find mappings between structured representations of data (specific data with a predefined format) and an outcome that is often seen as a prediction. These traditional machine learning algorithms often fail to create good mappings between data and an outcome when the data is of an unstructured or of complex nature (conglomeration of many varied types of data that are stored in their native formats) such as images, waveforms, video, etc. To tackle this problem, data scientists usually conduct data preprocessing and feature engineering processes to find data representations that are suitable for traditional machine learning of algorithms [79]

In deep learning, suitable representations of data (features) are found in a more automated way. Deep learning algorithms are capable of understanding concepts on their own, and this is referred to as representation learning [80]. This capability makes the algorithm less reliant on the prior domain knowledge of the person who performed the feature engineering process and more reliant on the neural network architecture chosen (number of hidden

layers, and the number of neurons in each layer, activation functions, etc.). Deep learning encompasses a set of algorithms that are multilayered and that can do automatic feature engineering. More precisely, deep learning algorithms attempt to solve the representation learning problem by creating hierarchical (layered) structures of concepts that can represent complex concepts from simpler or lower-level ones [79] [81].

In deep learning, each level learns to transform its input data into a slightly more abstract and composite representation. For example, in an image recognition application, the raw input may be a matrix of pixels. The first representational layer may abstract the pixels and encode edges. The second layer may compose and encode arrangements of edges. The third layer may encode a nose and eyes. The fourth layer may recognize that the image contains a face as illustrated in Figure 3-6. Importantly, a deep learning process can learn which features to optimally place in which level *on its own*.

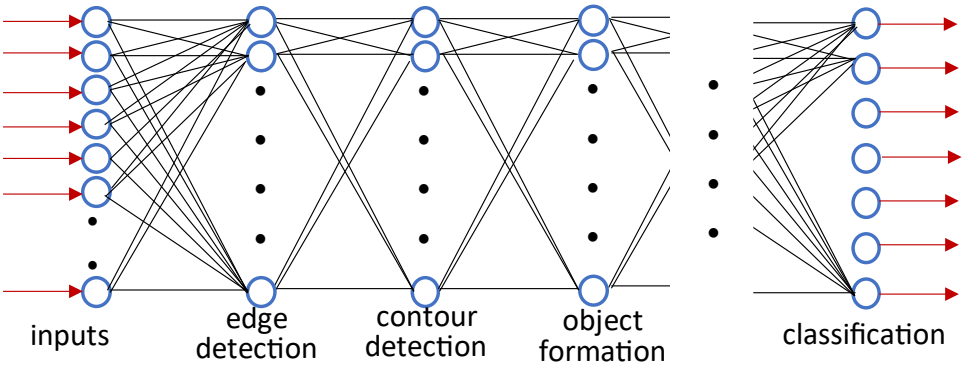


Figure 3-6: Conceptual representation of a deep neural network for image recognition

A deep learning model would require more data points to improve its accuracy, whereas a traditional machine learning model relies on less data given the underlying data structure. Deep learning models generally take longer time to train. Traditional machine learning algorithms have limited hyperparameters for tuning, but deep learning algorithms can be tuned in various ways.

3.5 Hybrid Intelligent Systems

A Hybrid Intelligent System (HIS) is a system that uses two or more of the techniques or algorithms belonging to artificial intelligence. HIS are generally made to solve complex problems by the combination of several techniques or algorithms [75].

HIS common combinations are Neuro-fuzzy Systems (FIS-NN), Evolutionary Fuzzy Systems (EC-FIS), Evolutionary Neural Networks (NN-EC) and Hybrid Evolutionary Algorithms. The General framework for hybrid architectures is shown in Figure 3-7.

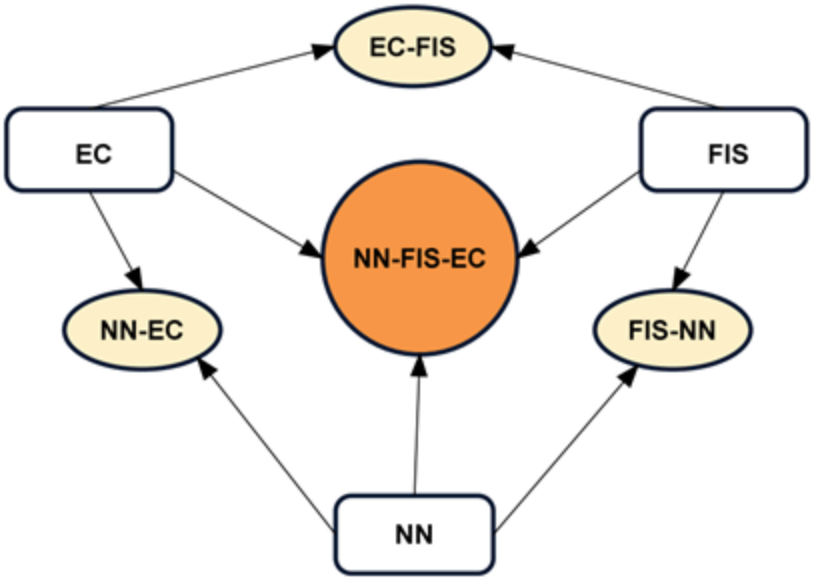


Figure 3-7: General framework for hybrid architectures [78]

Typical architectures for hybrid intelligent system are shown in the Figure 3-8.

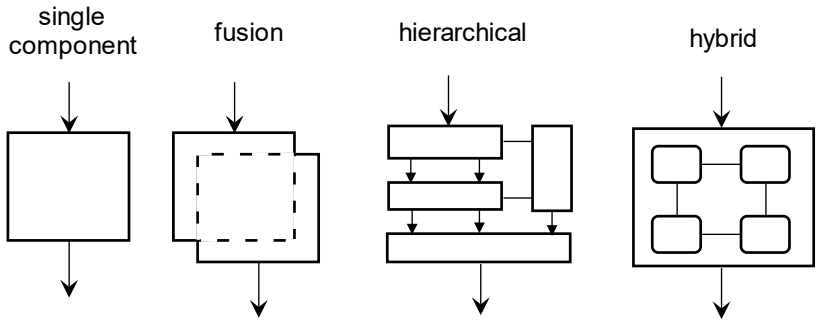


Figure 3-8: Typical architectures for hybrid intelligent system [82]

There are several ways to combine techniques or algorithms to create a hybrid intelligent system. Techniques are chosen based on the problem to be solved. Pattern recognition problems that include classifications usually use neural networks with fuzzy approaches to create Hybrid Fuzzy or Neural Systems to improve the performance and to make a more general capability according to variation of input data.

When finding an optimal solution of a problem, it is common to use a Hybrid Evolutionary System by combining evolutionary techniques or algorithms with fuzzy logic or expert systems. For example, expert rules can be defined with fuzzy logic in order to define the starting population of an evolutionary technique or algorithm in order to improve the convergence speed. In protection, it is common to use a combination of genetic algorithms

with fuzzy logic to optimize overcurrent protection and distance protection settings in single objective and multi-objective problems of protection coordination.

When the complexity of the problems requires several modules of task, every one of them with a complex level, the combination of techniques or algorithms becomes an intelligent system by the integration of more than two techniques or algorithms to be able to be successful. For example, to build a system capable to do a protection coordination study of a grid element like a power transformer, several tasks need to be done and can include, scenario analysis, topology recognition, expert rules application for settings, definition of verification faults and coordination rules, evaluation of proper coordination margins and an optimization method. All the problems therefore need the usage of several techniques or algorithms capable of performing classification, rules application, optimization and can be achieved by the combination for example of fuzzy logic, neural networks and evolutionary techniques or algorithms using a hierarchical architecture.

4 AI/ML for Protection & Control

4.1 Drivers

The existing protection and control applications have been working for many years assuming the conventional grid architecture, and they have performed exceptionally well thus far. However, modern power systems dynamics have changed, leading to protection and control facing new challenges to maintain/improve the network and asset performance metrics. Many engineers and researchers are exploring new approaches such as biology-inspired AI/ML-based solutions to the problems that cannot be properly resolved by conventional physics-based approaches. Increasing availability of large amounts of high-fidelity, rapidly sampled data, both in time domain and frequency domain is enabling the development and deployment of AI/ML-based solutions to challenging protection and control problems.

Some drivers leading the focus towards the Artificial Intelligence are:

- High penetration of Distributed Energy Resources
- Large scale inverter-based resources
- Growing need for advanced distribution protection and control applications including fault detection and location
- Wide area instabilities
- Power system reconfiguration
- Increased use of HVDC converters, FACTS devices and compensators
- Natural disasters
- Need for improving the efficiency in protection systems design and engineering
- Mitigating human errors

4.2 Limitations of existing P&C technology

Faults can occur on any electrical equipment/component due to various causes such as extreme weather, aging, impact due to repeated stress, adverse environmental conditions resulting in degradation of the insulation, internal/external damages during construction, etc. They result in outages of a specific asset or the entire network depending on the design of power system, severity, coverage of protection zone (e.g., line, transformer, reactors, buses, generators, loads).

Conventional power system protection schemes have been successfully applied to provide adequate protection functions under most power system conditions including normal operating conditions and various fault scenarios. However, there are conditions that conventional protection schemes had shown limitations such that existing schemes either could not provide adequate protections at all or the desired performance and/or dependability-security balance of the protection schemes could not be achieved. Conventional protection schemes work the best when the power system conditions that require protection actions are very different from all other conditions for which when no such actions are needed. Some protection schemes (e.g., out-of-step protection schemes, system integrity protection schemes) work better when the power system where the schemes are applied is not dynamic in terms of topology but may have difficulties to balance the dependability and security requirements if the topology of the power system is dynamic.

On the other hand, designing some protection schemes for dynamic or complex power systems is a combination of science and art. This means that there may not be any "standard" designs for such protection challenges. Solutions by different engineers could be different from each other, which can result in differences among the power systems where they are applied and/or the differences in personal experience and knowledge among the designers. The situation described is common, for example, in new transmission grid projects. New projects modify a part of the power system and they can create complex topologies because of weak short-circuit conditions with a high source impedance ratio (SIR), infeed, or special configurations like three-terminal lines. According to the regulations of some countries, in many cases, the projects do not replace the existing protection systems of existing elements, and the protection engineers would develop settings for new and existing protection systems to maintain proper protection coordination. Existing protection systems can present challenges if teleprotection is not available for distance protections or if only overcurrent relays are available. In these cases, regulatory fault clearing times are difficult to fulfill and even the performance of the backup protection coordination may be insufficient if redundancy is not available.

In protection coordination studies the major challenge when protection settings are calculated and verified is to achieve settings that are reliable for most operating scenarios of the power system (optimal settings). Power system expansion, new dynamics of equipment, and maintenance constantly add more complexity for finding the optimal settings of protection systems. Figure 4-1 shows the conventional focus of the reliability in protection systems.

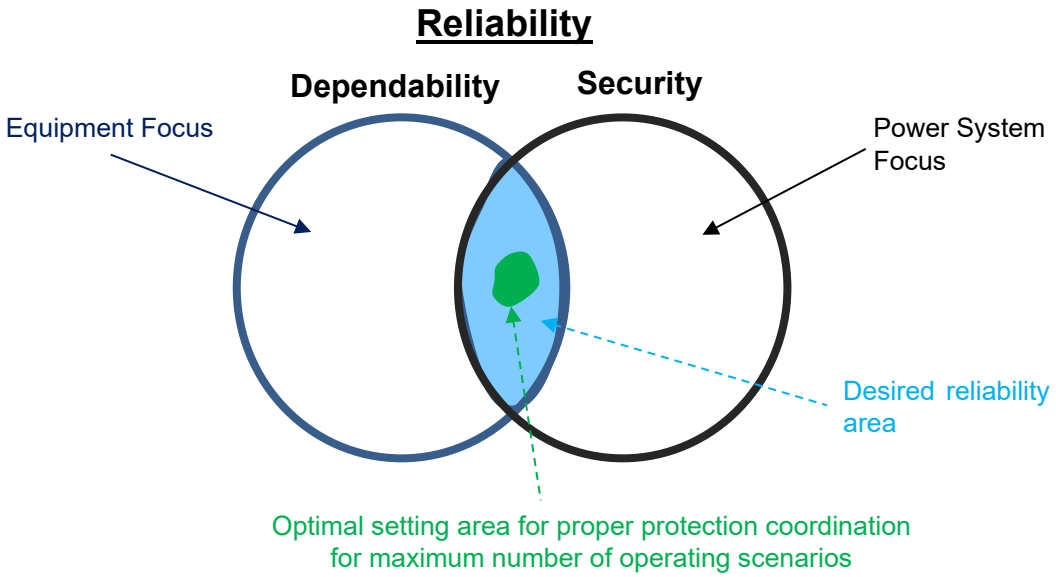


Figure 4-1: Optimal setting area for reliability

Microprocessor-based relays are basically an industrial computer with settings as determined by the protection algorithms. Pickup, dial, zone reach and operation time are the first to come to mind, but there are many more that require attention as well. Examples are polarization, phase selection, directionality, blocking logics, etc. Depending on the protection element or function, it may require a revision to optimize settings for proper protection coordination in all the possible operating scenarios.

In the presence of high inverter-based resources penetration, there are many challenges when using classic analytical methods to solve complex contingencies to determine optimal protection settings.

4.3 Opportunities for Artificial Intelligence in P&C

AI/ML can sometimes enhance existing power system protection and control functions that microprocessor relays currently perform, such as the AI application in addressing secondary arc example described in 5.1.1. It enables engineers to make more data informed decisions by learning and creating models from historical and near real-time data. By leveraging the data to make protection and control decisions, the engineers can efficiently adapt their protection and control approach to account for a dynamic grid whose electric properties and characteristics are changing.

Artificial intelligence could create efficiencies by adding value and insight into various stages of the protection and control process. The main stages of the P&C process that AI could impact are as shown in Figure 4-2:

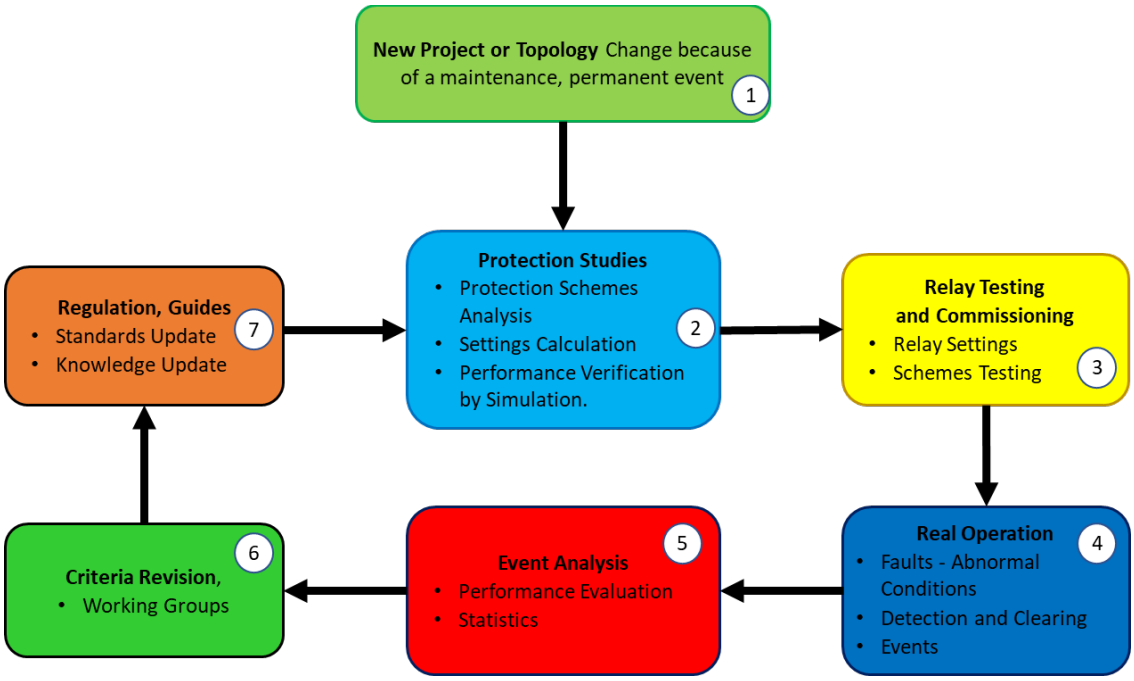


Figure 4-2: Main stages of the P&C process that AI could impact

Each of these tasks above provide an opportunity for some aspect of AI/ML to assist either the engineers or the P&C devices react faster, be more accurate, and become more efficient. Applications and use cases that benefit from AI/ML are discussed in the subsequent sections. These discussions emphasize how each application benefits from AI/ML and how it impacts the various stages. In general, AI/ML may supplement and aid protection engineers but will never replace them.

4.4 Considerations before using AI in protection

Although, as discussed in this section, there are opportunities for use of AI to protection, before using any new method to improve existing technology and practices, a philosophical as well as a technical evaluation of the pros and cons of the new method is in order. Principles and applications of power system protection have been refined and perfected over decades of practical experience and a number of revolutions in technology. Since success of protection improves safety of equipment worth millions of dollars, and more importantly, human lives, protection engineers have traditionally preferred transparent and relatively simple physics-based models.

Interestingly, pattern creation and classification, two foundational components of ML based methods that help build AI in a system, are already an integral part of protection. Practically every relay uses a feature or a combination of features (pattern), and a rule-based classifier. A feature is typically frequency-domain transformation of time-domain measurement – current, voltage, frequency, power, impedance, harmonic content, high-frequency content, etc. A pattern is created through the combinatory logic blocks provided in numerical relays. A classifier typically has two-classes: Fault & No Fault. Separation

plane for the classifier is a threshold value (for the feature) or a combination of threshold values (for the pattern). Values and combinations are determined through system analytics, physics and physics-based models, and experience. Here, experience is simply a representation of the human mind learning from past data. The difference is, the learning in ML applications happens (or experience is gained) through algorithms developed and refined by computer scientists. In applications using large sets of data, it has been shown that these algorithms "digest" the data better than a human mind can. ML based methods also tend to create and use quite complex patterns using mathematical transformations on raw data. Such transformations, when used to create a data-driven model of a physical process, often yield quantities that lose direct relationship to the physics underlying the process it is trying to model.

These observations lead to two philosophical questions that need to be considered before applying AI to improve or replace an existing physics-based method:

1. If AI uses the *same* features/patterns used by relays, why would it be able to create a more dependable and secure classifier?
2. If AI uses more complex and abstract features/patterns that have no transparent relation to the underlying physics, and thresholds (separation planes) are created simply by learning through data, why would it work better?

Extending the thought behind these questions, if a legacy protection is not working in certain system conditions, why would an AI based method work? If it uses the same patterns the legacy relay is using, or can use, applying learning to these patterns may not necessarily yield any better result, as the patterns have failed. A question about opaque patterns to solve this problem is valid in this case as well.

This discussion is against replacing physics-based time-tested methods with AI-based solutions without clear justifications, even if the existing methods fail occasionally. Unfortunately, a large number of papers published on this topic takes this approach. These papers also disregard the fact that power system protection is a system that encapsulates interdependent localized protection-schemes. Replacing one/few such schemes by a ML-based method may not reconcile with the holistic nature of power system protection [83]. A pragmatic approach could be to complement conventional physics-based approaches with biology-inspired ML approaches to provide maximum protection coverage, and optimize dependability and security based on asset/system requirement. This report will present application examples of AI that have shown to or could potentially help improve protections where underlying physical models are not precise or absent, leading to known weaknesses.

4.5 Specific areas of AI/ML P&C application

For this report, power system protection is classified into two categories: asset protection and network protection.

The asset protection function involves detection of faults or abnormal conditions, discrimination of faulted zones or devices, identification of fault types, and making trip or block decisions, sometimes adaptively.

The network protection functions such as System Integrity Protection Scheme (SIPS), Remedial Action Schemes (RAS) and Wide Area Monitoring, Protection and Control Systems (WAMPAC) requires recognizing rare critical conditions impacting the power systems by monitoring the diverse set of variables. These systems, upon detection of critical conditions, initiate emergency control actions.

Regardless of the focus, design of the protection function/algorithm, system analysis, selection of settings, and coordination often require trade-offs between the security (no false tripping), speed of operation, and the dependability (no missing operations). The more secure the relay is (both the algorithm and its settings), the more it tends to be less sensitive or operate slowly. And vice versa, the faster the relay is, the more it tends to operate falsely. The design and coordination of asset and network protection functions to achieve desired reliability and cost objectives may become a complex optimization problem in modern power systems. As the grid becomes more dynamic, the optimization objective becomes a moving target, and static settings and assumptions may not be sufficient.

4.5.1 Asset Protection

The limitations of classical asset protection techniques may arise from the extent of information available through the approach of using the same measurements used by existing relays (e.g., sampling frequency) and/or limitations of the relaying algorithms used for detection and discrimination. These limitations can manifest in the form of less-than-ideal speed and sensitivity to high impedance faults, or adaptability to various conditions such as high source impedance, series compensated lines, inverter-based sources, etc. To mitigate these limitations of the classical relaying principles, it is possible to improve and extend the range of measurements available for decision-making and improve the recognition process itself. For example, the high frequency transient signals contained in the input signals can be utilized to detect and discriminate high impedance, arcing and evolving faults, which are challenging to traditional protection function based on power frequency phasors. However, circuit laws applied to simplified models cannot characterize the relationship between the high frequency content and the fault location/severity. In such situations, AI/ML techniques can be used to train classifiers to make decisions based on various signatures contained in the high frequency components of the input measurements.

On a case-by-case basis, engineers can consider leveraging AI/ML models to augment the selectivity, security, and speed of classical relaying principles by

- Optimizing settings and utilizing hybrid approaches that use multiple relaying principles
- Automatically correcting CT and VT signals
- Employing non-conventional algorithms

4.5.2 Network Protection

With respect to network protection, significant advances have been made in the field of on-line security assessment technologies that can determine critical contingencies, transfer limits, and remedial measures. However, conventional numerical techniques based on

offline studies are usually time consuming and therefore are not always suitable for real-time applications. Also, these methods suffer from the problem of misclassifications or/and false alarms. Many studies show that AI/ML techniques can be applied to develop effective RAS and WAMPAC systems to detect critical or abnormal network conditions in real-time and propose or automatically activate remedial actions.

4.5.3 Post Event Analysis

Another context where AI/ML can be very helpful is post event analysis by using data analytics. Modern data mining techniques can be essential when it is required to analyze large data sets such as digital fault recorder data, synchrophasor data, utility meter readings, weather data, etc. and their combinations. Analysis of large structured or non-structured data sets is needed in many situations such as sequence of event analysis.

4.5.4 Summary of Potential P&C Areas for Application of AI/ML

Table 4-1 provides some examples of protection and related functions where application of AI/ML techniques could be helpful. The current intelligent electronics devices (IEDs) employ classical protection functions to achieve many of these functions with high degree of success, but there is some room for improvement in terms of speed, sensitivity, reliability, and adaptability as mentioned earlier. On the other hand, classical algorithms are inadequate or too complex to apply in real-time for some functions such as system stability prediction and control, islanding detection, controlled islanding, arc detection, etc. In these applications AI/ML can play a crucial role.

Table 4-1: Potential areas for application of AI/ML

Asset Protection	Network Protection
• High impedance fault detection • Evolving fault detection • Fault direction discrimination • Faulted segment identification • Detection of secondary arc extinction • Adaptive protection • Condition monitoring and fault diagnosis of transformers, reactors and capacitors • Detection of CT saturation, VT transients, and inrush conditions, as well as self-monitoring and diagnosis.	• System level monitoring and protection • Voltage stability monitoring and protection • Transient stability monitoring and protection • Islanding detection and protection • Frequency stability protection
Common to both asset and network protection	
• Alarm processing and diagnosis • Settings/coordination optimization, adaptive settings, and setting-less protection • Digital fault recorder data analysis • Sequence of event data analysis • Bad data detection and data conditioning	

5 APPLICATION ILLUSTRATIONS

The following example applications illustrate how AI/ML technology could be applied to resolve practical power system protection and control problems. The example applications are divided into two broad categories: in-use examples and high-potential examples.

When an AI/ML application has been implemented in a product or has been used to solve real-life power system protection and control problems, it is considered as an in-use example.

The inclusion of high-potential application examples could be somewhat subjective to the opinion of this working group: either there has been sufficient evidence, i.e., the current R&D work has shown some very good results, or the assessment of this working group has led us to believe that the application of AI/ML has a high-potential to provide a better solution over existing technologies.

The report only included the example applications that are known to or evaluated by this working group. There could be many other example applications that are currently in-use or considered as high-potential ones under the active R&D efforts.

5.1 Existing Applications

Two existing example applications of AI/ML technology for power system protection and control are described below. The first example application, where AI/ML technology is applied to detect secondary arc, has been implemented in a real product. The second example application, where AI/ML technology is applied to select the optimized relay settings, has been in use by a utility.

5.1.1 AI to Detect Secondary Arc

5.1.1.1 The problem: secondary arc detection

The problem of secondary arc detection is related to single phase tripping and autoreclosing. Single phase tripping and autoreclosing is used to clear temporary single phase to ground faults. In case of a single phase to ground fault, the faulted phase is tripped from both ends of the line and reclosed after a short time, e.g., typically less than 1 sec (30-50 cycles). In most cases the fault has disappeared during this time and the reclose is successful.

Single phase tripping refers to opening only the faulted phase at both ends of the line. Therefore, no current can flow via the faulted phase directly into the fault anymore. The primary arc extinguishes.

Unfortunately in some cases a secondary arc appears, which is fed by the two healthy phases via capacitive coupling like shown in Figure 5-1. The voltage $\underline{U}_M$, measured at the disconnected phase is characterized by the **ohmic nonlinear behavior** of the secondary arc R_{ARC}.

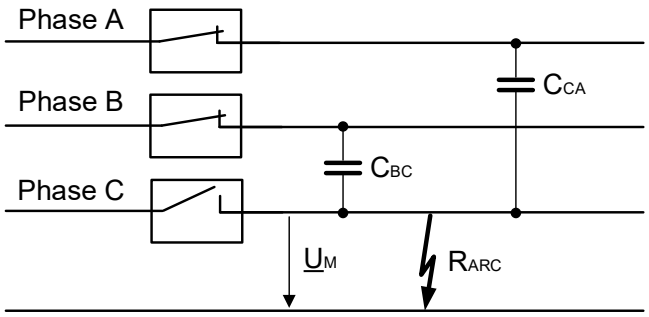


Figure 5-1: Simplified diagram showing the secondary arc after single phase tripping

If the secondary arc is extinguished, the equivalent circuit is changing to a different model like shown in Figure 5-2. The voltage $\underline{U}_M$, measured at the disconnected phase after extinguishing of the secondary arc is characterized by the **linear capacitive behavior** of the phase to ground capacitance C_{CG} of the open conductor.

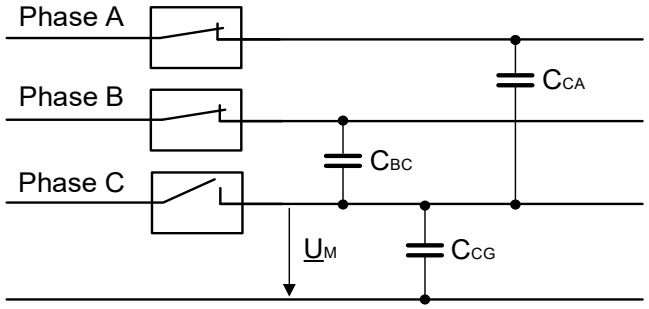


Figure 5-2: Simplified diagram showing the secondary arc extinction after single phase tripping

A reclose at presence of the secondary arc is not successful as shown in Figure 5-3. After reclosure, the fault persists which leads to a final trip of the protection. This unsuccessful reclosure could be prevented using a secondary arc detection function.

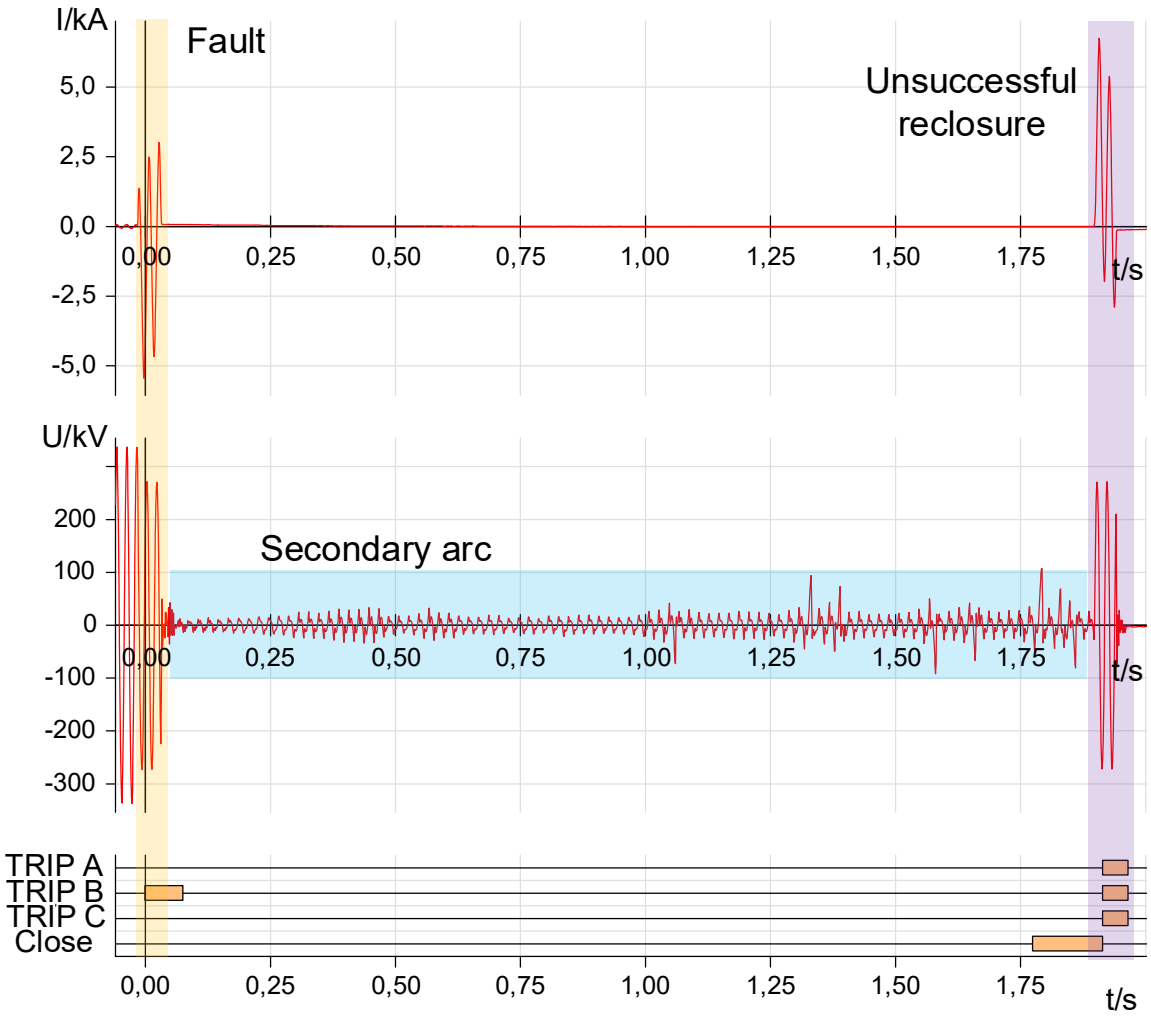


Figure 5-3: Unsuccessful reclosing if secondary arc is not extinguished after single phase tripping

A successful reclosure after extinguishing of the secondary arc is shown in Figure 5-4. After tripping the line, the fault current disappears. At the same time the voltage starts the typical nonlinear behavior of arcing. Later the secondary arc extinguishes, and the voltage is changing to a linear capacitive behavior. After successful reclosing both voltage and current go back to normal conditions. In this case the single-phase dead time could be reduced from 1.2 s to 0.5 s, if the extinction of the secondary arc would be detected.

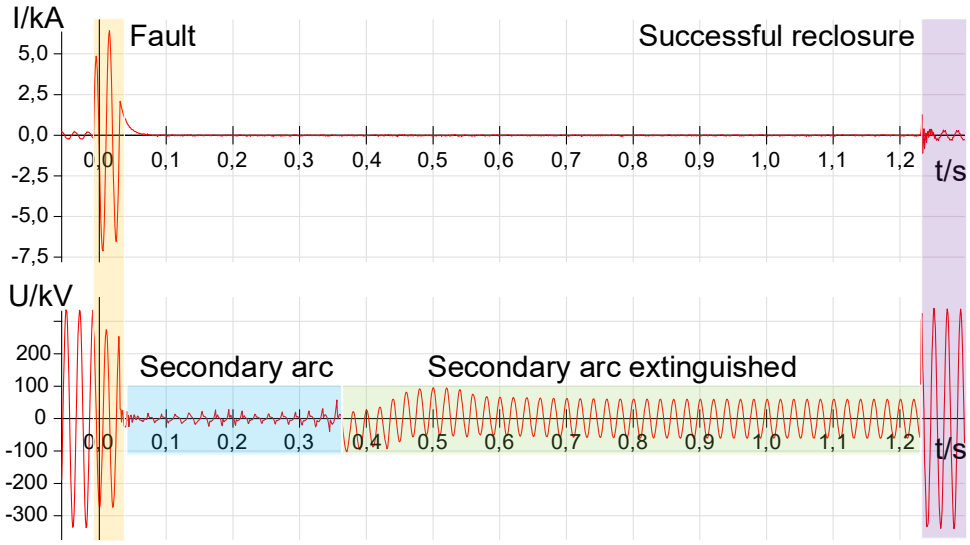


Figure 5-4: Successful reclosing if secondary arc is extinguished after single phase tripping

5.1.1.2 Implemented solution

As described in [84], there are several algorithms to detect the secondary arc for instance based on harmonics or phase angle of the open-phase voltage. All these algorithms try to distinguish between the ohmic, non-linear behavior of the secondary arc and the capacitive behavior of the voltage if the secondary arc is extinguished.

The behavior of arcing however can be quite different depending on several factors. That is why another algorithm was added based on a neural network to detect the presence of the secondary arc.

Finally, the secondary arc detection function uses several criteria to detect the secondary arc as shown in Figure 5-5.

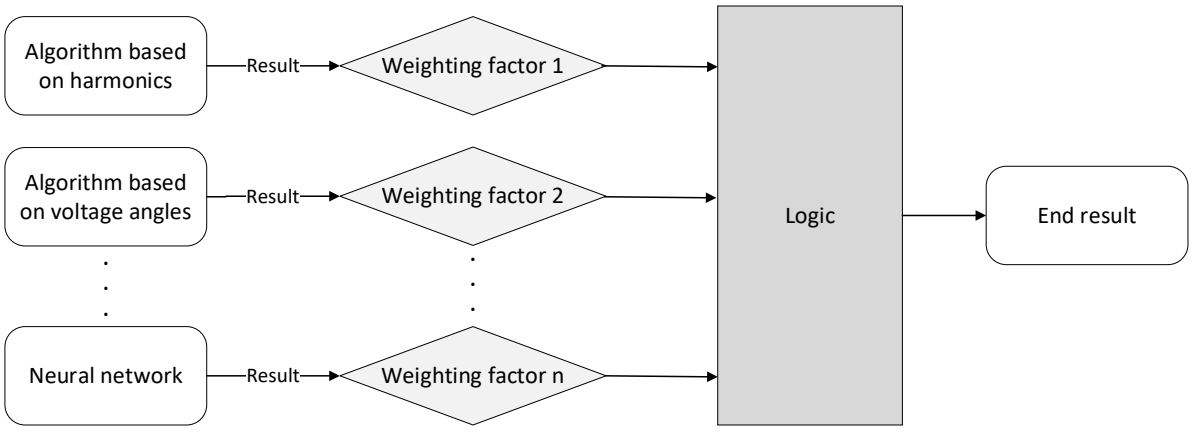


Figure 5-5: Structure of the secondary arc detection function

As shown in Figure 5-5 the neural network was added as one criterion to detect a secondary arc.

5.1.1.3 Training of the neural network

A neural network was added as a criterion to detect a secondary arc based on the waveform of the open-phase voltage. A neural network results in an optimal algorithm for classifying between “secondary arc” and “no secondary arc” detection if it was trained correctly with a representative set of data.

The neural network shown in Figure 5-6 as example contains 9 neurons in three layers. In a learning process these neurons were optimizing their weight factors. To achieve a good result a lot of different training data is necessary. The training data consists of fault records with an analog channel representing the open-phase voltage and a binary signal representing the information about the presence of the secondary arc.

Most of the training data was computed by network simulation programs, simulating different conditions of secondary arc. These training data was augmented by a large amount of fault records from “real” secondary arcs, received from different customers around the world.

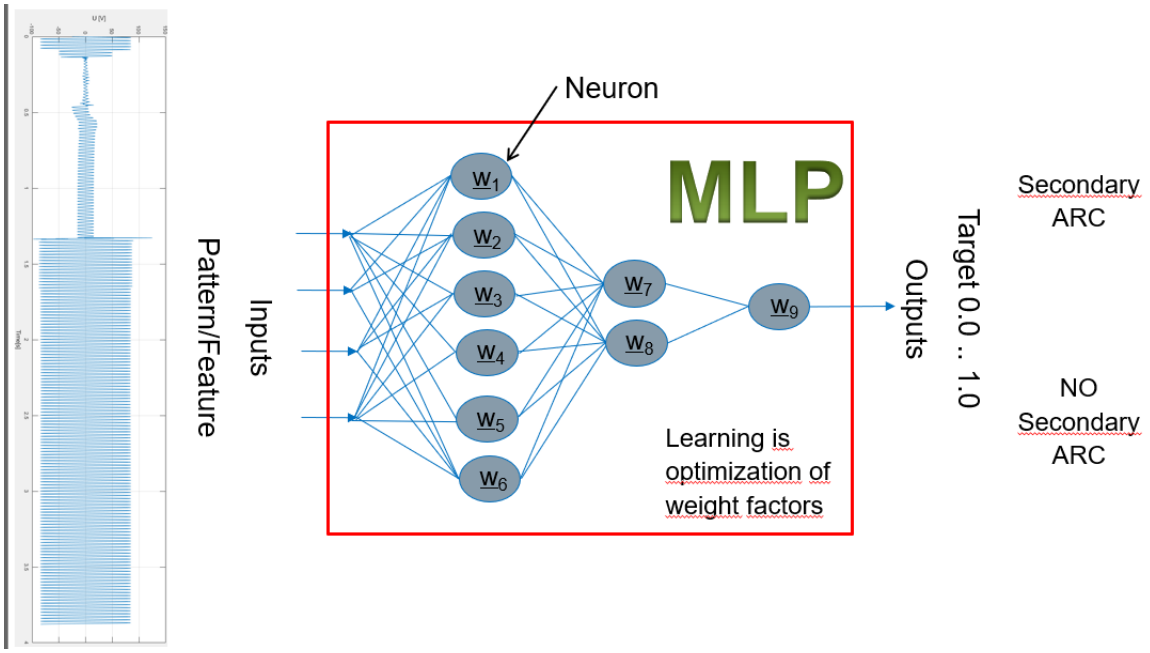


Figure 5-6: Learning process of the neural network used for secondary arc detection

5.1.1.4 Results

The following three figures are illustrative of the successful function of secondary arc detection of the implemented neural network. The figures are based on fault records showing the voltage of the open phase in blue and the output signal of the neural network in red color.

Figure 5-7 shows a secondary arc which is extinguished approximately 300 ms after the single-phase tripping. The neural network detects the state change very fast and precise.

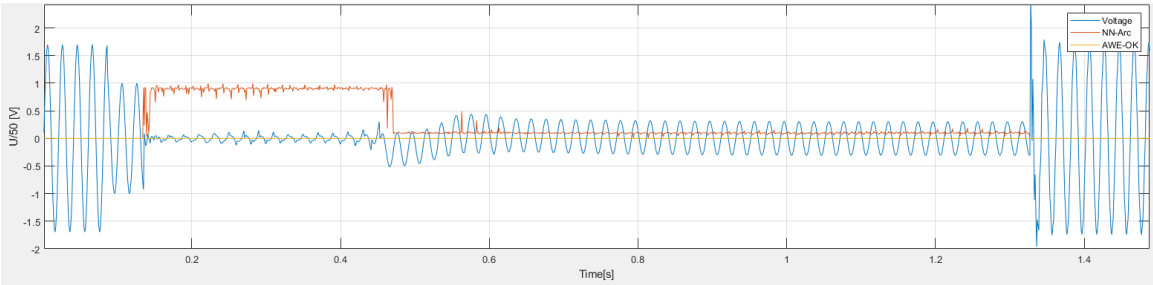


Figure 5-7: Detection of secondary arc extinguishing after single phase tripping

Figure 5-8 is showing a secondary arc which is changing its shape but does not extinguish during the single-phase dead time. The output of the neural network is stable and indicates the presence of the secondary arc during the whole time.

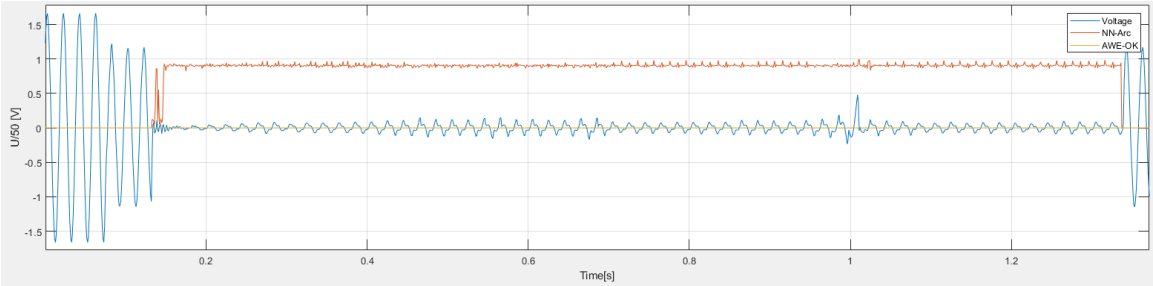


Figure 5-8: Secondary arc not extinguishing during the dead time of autoreclosure

Figure 5-9 shows a secondary arc which is re-arcing after extinguishing the first time. The neural network detects the first extinguishing of the secondary arc and the re-arcing very fast and precise. After re-arcing, the output of the neural network is correct but not as stable as previous example.

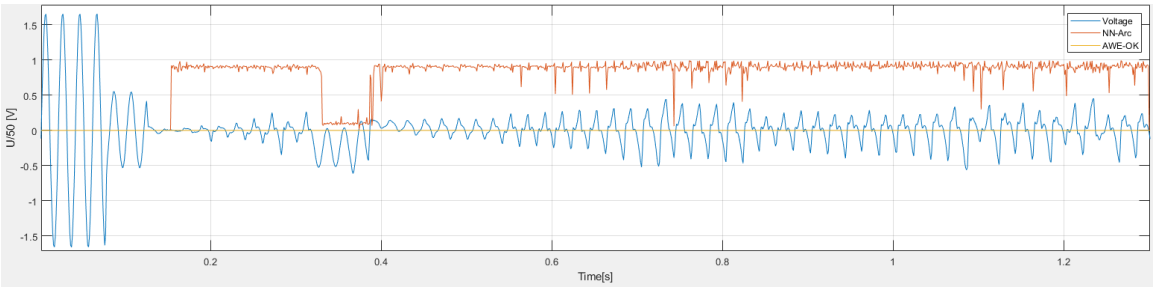


Figure 5-9: Detection of re-arcing after extinguishing of secondary arc

Due to the concept of using several criteria to detect the presence of the secondary arc the risk of a malfunction in case of a secondary arc which is not detected with the neural network is minimized.

5.1.2 Hybrid Intelligent Model for Protection Settings Optimization

5.1.2.1 The Problem: Finding Optimal Settings for Distance Protection

Distance protection settings are difficult to optimize because several electrical conditions impact its performance, sometimes in competing manner. SIR, load flow, topology, type of fault, fault resistance, zero sequence mutual coupling of paralleled lines, infeed of 3-terminal lines can modify the impedance seen by protection relays and cause an overreach or underreach problem. Figure 5-10 illustrates the challenges in finding optimal distance relay settings.

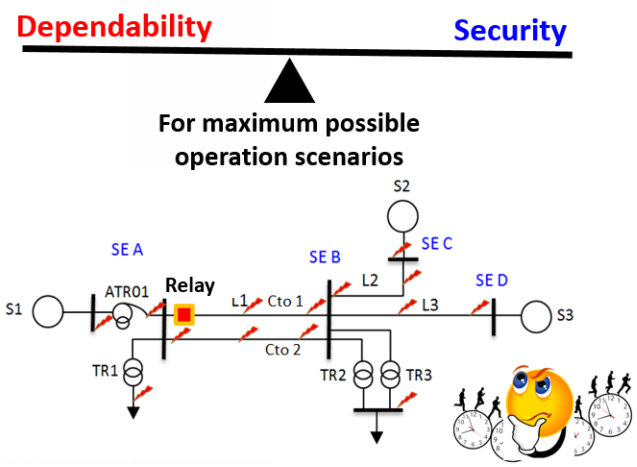


Figure 5-10: Problem of Finding Optimal Distance Relay Settings

From the point of view of a power system protection engineer, the problem of finding optimal settings in protection studies for distance protection means to find the reach values of every zone, that is inductive reach and resistive reach, for reliable operation of the relay in all the possible scenarios considering at least one level of contingency (N-1 criteria). It is a highly time-consuming task to get a set of optimal settings.

5.1.2.2 Hybrid Intelligent System for Distance Protection Settings Optimization

A Hybrid Intelligent System (HIS) for distance protection settings optimization was proposed, implemented and tested and the work and the results are reported in [85]. The techniques used in the HIS model were Constructive Methods (sequential building of solutions), Fuzzy + Conventional Logic, and Differential Evolution (DE). Figure 5-11 shows the structure of the HIS that contain a number of functional modules.

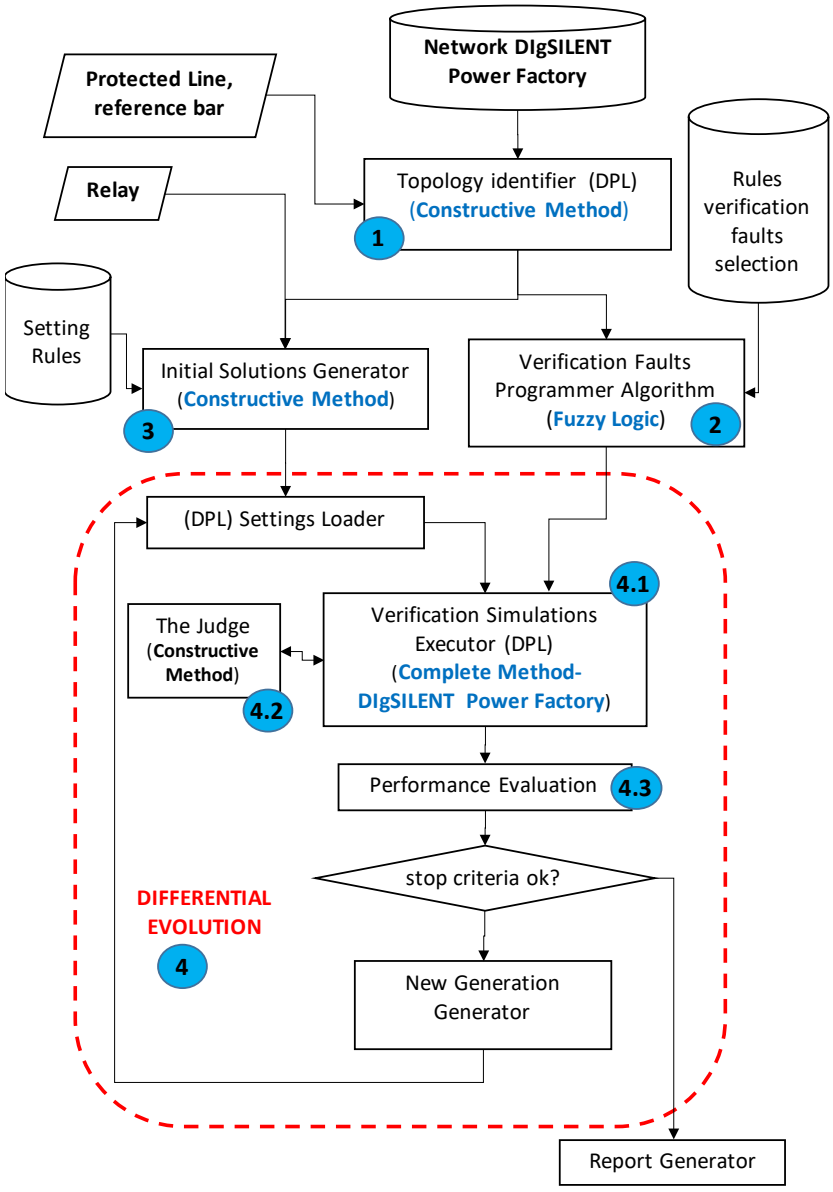


Figure 5-11: HIS for Distance Protection Settings Optimization

The HIS was designed to automate the whole process of calculation, verification and optimization of distance protection relay settings by the execution of faults considering normal operation scenarios and N-1 contingencies.

First part of the HIS considers automatic topology identification based on object-oriented programming, which is labeled as Topology Identifier (1) in Figure 5-11. The topology identification is used to program the faults to be applied to evaluate performance of the settings. Programming of faults uses a hybrid logic that uses both fuzzy and conventional logic. Then the module labeled Verification Faults Programmer Algorithm (2) in Figure 5-11 generates a set of faults according to the topology around the protected line, for evaluating the performance of the settings.

After Verification Faults (VF) set is build, typical distance protection settings are calculated by the Initial Solutions Generator (3) in Figure 5-11 as the initial population for the optimizer algorithm that is based on differential evolution. Solution structure is shown in Figure 5-12.

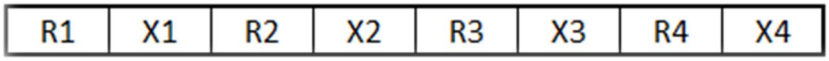


Figure 5-12: Solution structure for zones 1, 2, 3 forward and 4 reverse

The relay coordination problem is typically formulated as mixed integer non-linear programming (MINLP). Most research efforts had been focused on overcurrent relays. Considering ranges involved in variables in distance relay setting problem, differential evolution was chosen over neural networks and vector support machines because of generalization required to permit several network types (several sizes related to nodes and topologies). The differential evolution was also chosen over genetic algorithms and swarm techniques because this work was focused on only one relay and has several continuous variables with known ranges. Coordination with other relays is done via constraints.

To find the global optimal, a modified version of the DE algorithm was implemented in HIS, as shown in Figure 5-13.

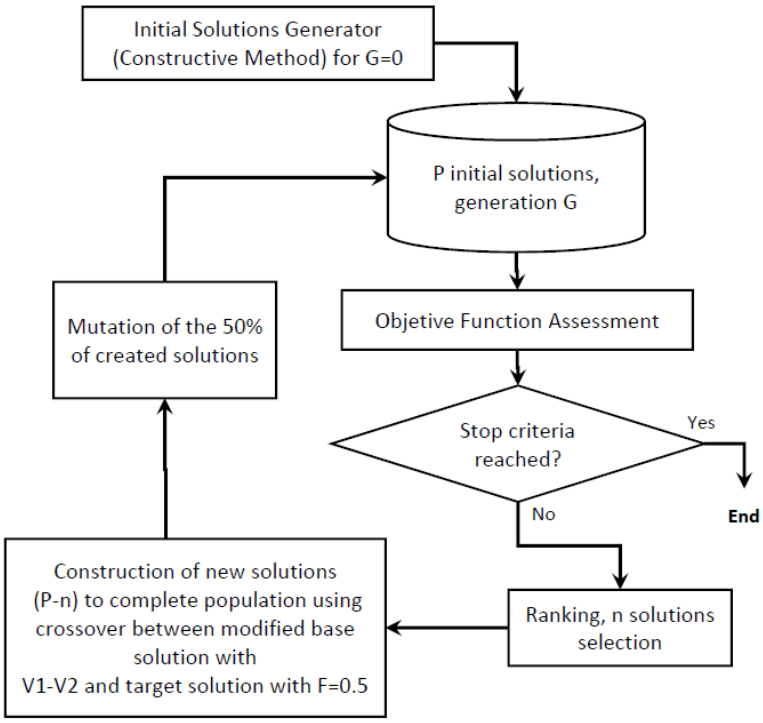


Figure 5-13: Differential evolution modified algorithm of HIS

An innovative concept to quantify performance of a distance protection relay setting was developed and implemented in the module labeled as The Judge (4.2) in Figure 5-11. The basic formulation of Judge function is shown in Figure 5-11 and the performance evaluation function (FDR) is explained in detail below.

The basic idea is to give points according to the performance of every fault (FV is the set of faults to be evaluated) considering the ideal expected result and the simulated operation based on the reliability concept. The FDR function has two components, DA and P. DA component gives points to every fault according to expected result (REOF) and the operation time of the relay for the fault being evaluated (TOF). Parameters k1 to k5 are selected according to the performance priority (dependability of security). P component gives to the fault a general weight factors according to how relevant and frequent is the fault. The factors consider fault location FPUF, fault type FPTF and fault resistance FPRF.

The function for the evaluation of the performance of the relay is:

$$FDR = \sum_{i \in FV} DA_i(\mathrm{REOF}_i, \mathrm{TOF}_i) \times P_i(\mathrm{FPUF}_i, \mathrm{FPTF}_i, \mathrm{FPRF}_i)$$

where:

FDR: performance function
FV: the set of faults to be evaluated
DA_i: Component for evaluation of relay at the fault i
P_i: Component according to fault characteristics

Component DA_i is calculated as follow:

$$DA_i = \begin{bmatrix} if\ TOF_i \in REOF_i \rightarrow k_1\ (\text{according to expected}) \\ if\ TOF_i > REOF_i \rightarrow k_2\ (\text{delayed trip}) \\ if\ TOF_i < REOF_i \rightarrow k_3\ (\text{too fast trip}) \end{bmatrix} \times \begin{bmatrix} if\ TOF_i < NO \ \wedge\ REOF_i = NO \rightarrow k_4\ (\text{undesired trip}) \\ if\ TOF_i \in NO \ \wedge\ REOF_i < NO \rightarrow k_5\ (\text{trip omission}) \\ \text{in other wise} \rightarrow 1 \end{bmatrix}$$

Where:

$REOF_i$: Expected Range of Operation of the relay for the Fault i
TOF_i : Operation Time of the relay at Fault i
k_1, k_2, k_3, k_4, k_5: factors selected according to the focus (dependability or security). Further details in [85].

Component P_i is calculated as follow:

$$P_i = FPUF_i \times (FPTF_i + FPRF_i)$$

$FPUF_i$: Fault location weight factor for fault i: $= \begin{bmatrix} \text{Internal fault} \rightarrow 1 \\ \text{External fault} \rightarrow 0.7 \end{bmatrix}$

$FPTF_i$: Fault location weight factor for fault i: $= \begin{bmatrix} \text{single fault to ground} \rightarrow 0.9 \\ \text{double} - \text{phase} \rightarrow 0.25 \\ \text{double} - \text{phase to ground} \rightarrow 0.4 \\ \text{three phase} \rightarrow 0.3 \end{bmatrix}$

$FPRF_i$: Fault resistance weight factor for fault i: $= \begin{bmatrix} 0 - 10\ \Omega & \rightarrow & 1 \\ > 10 - 20\ \Omega & \rightarrow & 0.9 \\ > 20 - 40\ \Omega & \rightarrow & 0.8 \\ > 40 - 70\ \Omega & \rightarrow & 0.5 \\ > 70 - 100\ \Omega & \rightarrow & 0.3 \\ > 100\ \Omega & \rightarrow & 0.2 \end{bmatrix}$

The optimization model used is shown in Figure 5-14. The key concept is to find the best setting according to the priority of dependability and security. For prioritizing the dependability, performance is calculated using parameters focused on dependability to promote tripping over undesired tripping. On the other hand, to prioritize security, performance is calculated using parameters that prioritize points to avoid undesired operations.

The optimization model is:

$$Max\ FDR_1 \times \left(1 - dF_{seg}\right) + FDR_2 \times dF_{seg}$$

Where:

FDR_1: performance of the relay for a set of faults FV, for parameters k_1, k_2, k_3, k_4, k_5 with dependability focus.
FDR_2: performance of the relay for a set of faults FV, for parameters k_1, k_2, k_3, k_4, k_5 with security focus.
dF_{seg}: User input factor to define security vs dependability priority.

$$Max\ FDR_1 \times (1 - dF_{seg}) + FDR_2 \times dF_{seg}$$

Dependability Security

dF_{seg} parameter [0,1] for priority of security vs dependability

FDR_1 with parameters k_1, k_2, k_3, k_4, k_5 focused on dependability.

FDR_2 with parameters k_1, k_2, k_3, k_4, k_5 focused on security.

Figure 5-14: Optimization Model used in the HIS

5.1.2.3 Test Results

Figure 5-15 shows the test system used for initial testing of the HIS to prove the concept. Conventional settings are optimized according to the electrical conditions leading to a reduction in zone 1 reach setting to avoid undesired tripping.

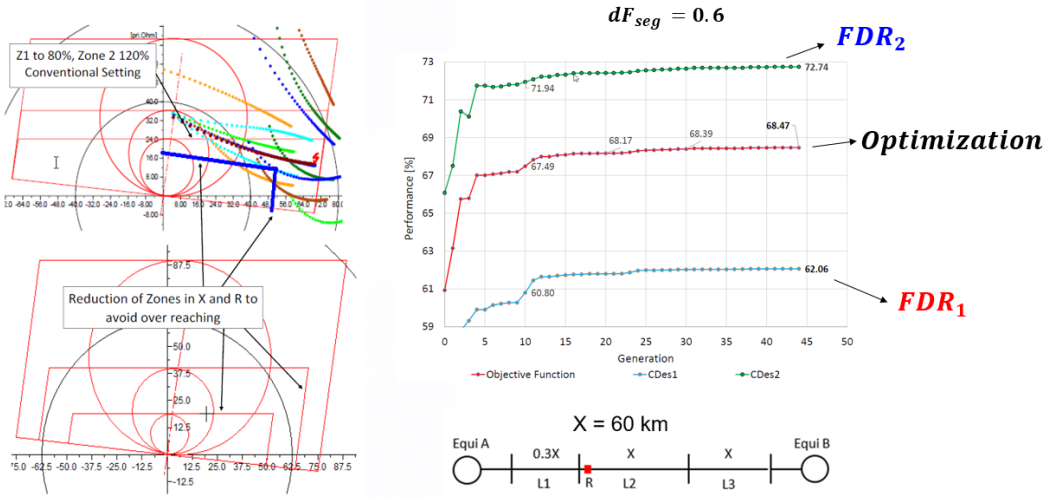


Figure 5-15: HIS test system results

5.1.2.4 Real Life Application Example

During a blackout in a section of the Colombian Power System, caused by a double line-to-ground bus bar fault at the substation JM in the level of 115 kV, the 115kV line relay at PS looking to JM did not trip. Moreover, readings showed apparent impedances that

differed from the line length parameter of the protection study. The utility owner of the PS – JM 115kV line corrected the line length to 6 km and thus a new protection coordination study was needed. The new length presented a SIR of 25 at the 115 kV JM terminal at PS. Short circuit level at substation PS is 6.4 kA and at substation JM is 7.3 kA. The high SIR condition and very short length of the line presented a settings challenge. The protection scheme available for the line bay PS to JM 115 kV was ANSI/IEEE 21, ANSI/IEEE 67/67N without teleprotection. Figure 5-16 shows the topology of the grid.

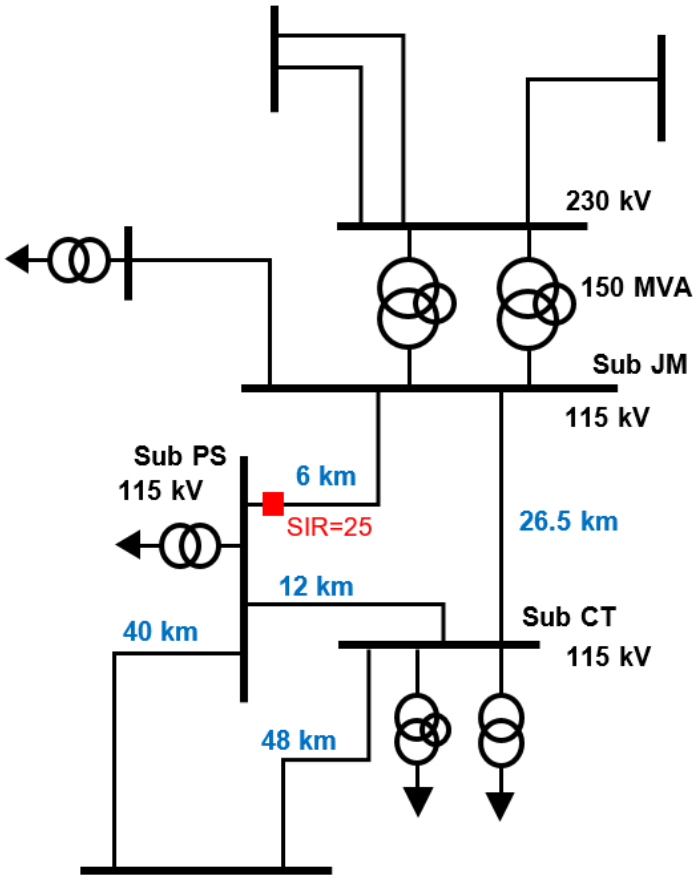


Figure 5-16: Real life application example grid

The HIS was used as a support for the decision about the best settings of the line bay PS to JM 115 kV. The best solution after 22 generations is shown in Table 5-1 together with one of the initial solutions based on conventional criteria. The percentages in settings of Table 5-1 are with respect to the impedance values of the line for inductive reach. For resistivity reach, the percentage is with respect to the minimum load impedance, considering 450 A of current capacity.

Table 5-1: Comparison Optimized vs Conventional Setting line bay PS to JM 115kV

Setting	Optimization Result [Ω]	[%]	Conventional Criteria [Ω]	[%]
Zone 1 X	1.85	71.1	2.23	85.7
Zone 1 R	47.56	32.2	63.7	43.2
Zone 2 X	6.19	237.8	3.41	131.0
Zone 2 R	84.93	57.6	63.7	43.2
Zone 3 X	8.94	343.4	16.43	631.1
Zone 3 R	20	13.6	63.7	43.2
Performance [%]	50.2		27.8	

After the analysis of the results of the optimal solution, it was identified an incursion in zone 1 for 4 to 6 ohms single phase to ground fault at the remote busbar (Sub JM) of the bay under study. Figure 5-17 shows the incursion of the single line to ground fault of 4 ohms.

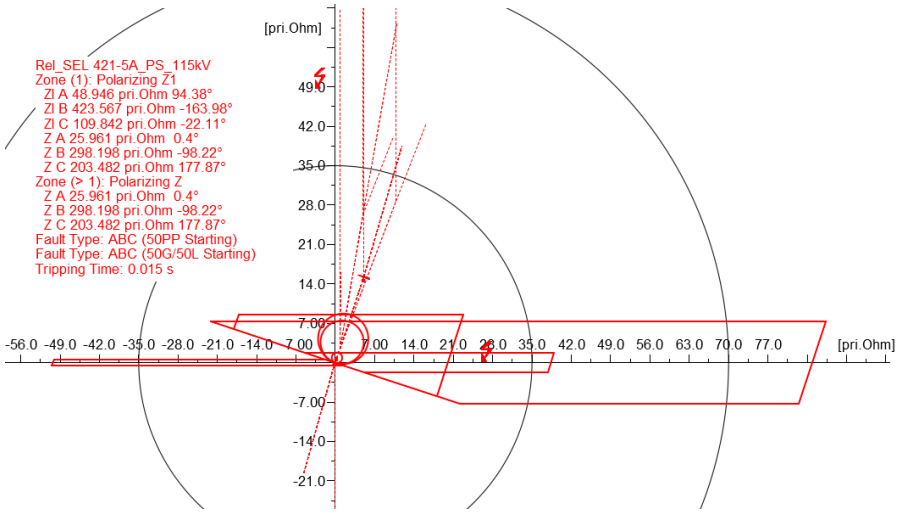


Figure 5-17: Zone 1 incursion problem for of real-life application example grid

The incursion shows the balance of the HIS that tried to achieve dependability and security. The optimal settings cover faults between 5 ohms or below to be detected along the protected element. The conventional solution in this case has several zone 1 incursions for faults in the adjacent elements and it does not cover all faults in the protected element with a fault resistance between 0 to 5 ohms, leading to a low performance of the settings as the Table 5-1 shows. The final decision considered the settings proposed by the HIS including a timing of zone 1 of 100 ms. Zone 3 had an important reduction due to the incursion of faults in the busbar of 230 kV substation JM. Figure 5-18 shows the best solution for every generation.

NoGen	idSol	CDesT	CDes1	CDes2			
0	1	36.460876	27.166849	42.656895	1-4.95|1.61|4.95|2.67|4.95|11.95|4.95|0.20	Inicial-Construct	
1	16	45.217319	29.873495	55.446537	16-20.00|1.98|20.00|2.97|20.00|8.71|20.00|0.59	4V1-9V2-1VB-2VT_C	
2	44	47.833385	35.496536	56.057955	44-63.70|1.96|63.70|4.94|20.00|11.20|63.70|0.59	16V1-19V2-26VB-15	
3	69	48.800915	34.498388	58.335932	69-63.70|1.96|63.70|4.55|20.00|8.71|63.70|0.59	16V1-41V2-50VB-44	
4	77	49.182224	34.808313	58.764835	77-74.69|2.05|63.70|5.13|20.00|8.71|41.85|0.59	53V1-69V2-44VB-58	
5	108	49.293072	35.690708	58.361315	108-71.95|1.87|84.93|5.71|20.00|9.33|37.15|0.59	71V1-80V2-79VB-77	
6	115	49.494923	35.142453	59.063233	115-84.93|1.83|63.70|5.58|20.00|8.22|58.83|0.59	108V1-71V2-103VB-	
7	145	49.585888	35.397305	59.044942	145-84.93|2.25|84.93|5.54|20.00|8.83|50.28|0.59	108V1-119V2-113VB	
8	160	49.856815	35.619399	59.348428	160-45.24|2.25|84.93|5.58|20.00|8.50|37.15|0.59	150V1-115V2-145VB	
9	160	49.856815	35.619399	59.348428	160-45.24|2.25|84.93|5.58|20.00|8.50|37.15|0.59	150V1-115V2-145VB	
10	210	50.030102	35.851441	59.482538	210-41.70|2.21|84.93|6.20|20.00|8.83|31.01|0.59	190V1-160V2-189VB	
11	229	50.134266	35.960762	59.583265	229-45.24|2.25|84.93|6.19|20.00|8.99|65.07|0.59	205V1-207V2-208VB	
12	239	50.134266	35.960762	59.583265	239-45.24|2.25|84.93|6.19|20.00|8.99|55.19|0.59	225V1-210V2-229VB	
13	256	50.137440	35.930369	59.608823	256-45.24|2.02|84.93|6.19|20.00|8.74|55.19|0.59	247V1-245V2-229VB	
14	275	50.163078	35.949219	59.638983	275-46.57|2.02|84.93|6.19|20.00|8.74|55.19|0.59	265V1-255V2-229VB	
15	297	50.163078	35.949219	59.638983	297-47.45|1.92|84.93|6.19|20.00|8.74|57.66|0.59	256V1-289V2-275VB	
16	319	50.163078	35.949219	59.638983	319-46.57|2.02|84.93|6.19|20.00|8.90|55.19|0.59	297V1-305V2-299VB	
17	347	50.163078	35.949219	59.638983	347-47.67|2.02|84.93|6.19|20.00|8.90|55.19|0.59	297V1-329V2-275VB	
18	369	50.163078	35.949219	59.638983	369-46.57|2.16|84.93|6.19|20.00|8.82|55.19|0.59	345V1-319V2-339VB	
19	387	50.205494	36.016831	59.664602	387-46.57|1.85|84.93|6.19|20.00|8.94|57.66|0.59	361V1-369V2-363VB	
20	393	50.205494	36.016831	59.664602	393-47.56|1.85|84.93|6.19|20.00|8.94|53.36|0.59	385V1-378V2-383VB	
21	425	50.205494	36.016831	59.664602	425-46.57|1.85|84.93|6.19|20.00|8.94|58.73|0.59	403V1-405V2-407VB	
22	447	50.205494	36.016831	59.664602	447-47.56|1.85|84.93|6.19|20.00|8.94|54.98|0.59	393V1-425V2-387VB	

Figure 5-18: HIS Execution final solution

5.2 Emerging Applications

In the following, example applications, where AI/ML technology may have a high potential to these application areas, are described.

5.2.1 High-Impedance Fault Detection with Artificial Neural Networks

High-impedance fault (HIF) poses unique challenges to conventional protection schemes operating on the premise of significant fault current availability, as it typically creates little to no fault current. Depending on the HIF type, e.g., downed conductors, persistent arcing, and intermittent arcing, the fault signature of each HIF type can be very different from each other. Therefore, it is difficult to detect HIF using conventional protection methods.

Although HIF rarely causes secondary equipment damage, it is critical to detect and clear the HIF promptly, to reduce safety hazards to the public and surrounding environment, e.g., wildfire, electrical shock, etc.

AI approaches, such as ANN, have been applied to address this technical challenge as early as the 1990s. Figure 5-19 presents a typical ANN structure. What is important when using ANN is:

- Selecting what the inputs to the ANN will be
- Selecting the ANN architecture
- Using the right data for the training
- Being as close as possible to reality

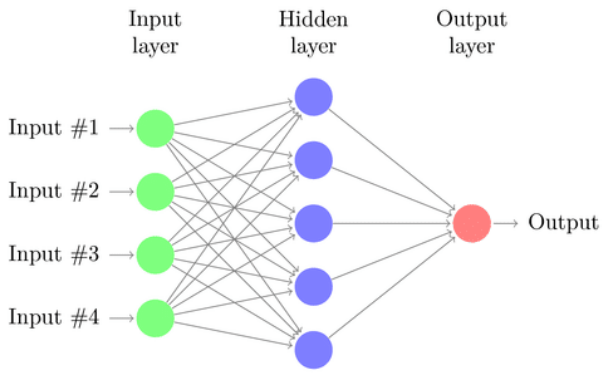


Figure 5-19: Typical ANN structure.

As an example, this section describes an R&D project using ANN based HIF detection method that was carried out at New York State Electric & Gas, Binghamton, NY in 1991. It successfully demonstrated the usability of ANN for high impedance fault detection. As a result of the project, two patents [86] [87] were issued to the team.

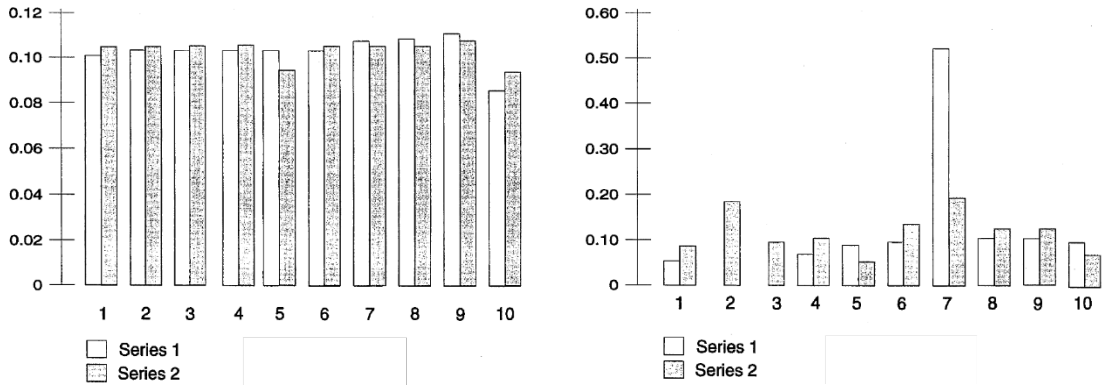
The method first performs pre-processing of the analog signals representing phase currents. They are provided to the inputs of an analog-to-digital (A/D) converter operating with a sampling rate of 80 samples/cycle. Digitized representations of the phase currents are applied to a pre-processor via a data bus. The pre-processor performs several mathematical and data formatting operations and then the pre-processed data is applied to the input of a trained ANN.

Training the ANN is necessary to establish the weights to be used to classify events occurring on the network as HIFs or normal switching events. Training was accomplished by providing multiple sets of known data representing both HIF and normal network conditions to the ANN along with the correct labels corresponding to each data set. Data sets may be compiled using computer simulation techniques or may consist of actual field-collected data corresponding to both fault and no-fault conditions. Actual field-collected data and superposition combinations of field data were used to train the ANN. Over 300 sets of field-collected and superposition data were used for the training. A typical training data set consisted of data points representing 10 seconds of sampling time on a real or simulated network.

The backpropagation of errors method was the learning technique chosen to train the ANN. A learning rate of 0.05 and a momentum (a method of changing weights based on a previous weight) of 0.1 were chosen. The data processing also included the identification of zero-crossing points in the data for each phase conductor. The maximum and minimum currents for each cycle are determined.

It has been found through experimentation with both simulated and field recorded data that calculating the first derivative is essential to the process of accurately detecting HIFs. For purposes of illustrating the importance of the first derivative in detecting a high-impedance fault, refer to Figure 5-20 (a) and Figure 5-20 (b). The bar graphs of ANN output (relative

value or relative HIF probability) vs. time are shown for a 10-second period for an actual HIF condition occurring on a single phase of a 13 kV three-phase distribution line. For this fault, the phase B leg of the distribution line was dropped onto ice. Series 1 data shows the faulted condition while Series 2 data shows the line under no fault conditions. In Figure 5-20 (a), the first derivative has not been calculated and as may be seen, there is little difference in the relative value of Series 1 or Series 2 data. In Figure 5-20, identical data is plotted. However, the first derivative has been calculated, and consequently there is pronounced difference in the output of the ANN in response to the HIF (relative value of the Series 1 data).



(a) Without rate of change calculation (b) With rate of change calculation

Figure 5-20: ANN output – relative HIF probability.

In conclusion, ANNs have been around for more than half a century. With today's huge amount of data in digital substations and the integration of distributed inverter-based energy resources they will play a very important role in the future protection and control systems. This HIF detection ANN application project 30 years ago demonstrated how useful such methods can be for detecting conditions that are difficult to protect using conventional methods. The above is only one of the examples – Another application of ANN for high impedance fault detection can be found in [113].

5.2.2 Use of synchrophasor and ML for Early Detection of Long-term Voltage Instability

5.2.2.1 Introduction

Voltage stability refers to the ability of a power system to maintain steady voltages close to nominal value at all buses in the system after being subjected to a disturbance, and the phenomenon is divided into two categories: (i) Short-term voltage stability which involves dynamics of fast acting load components such as induction motors, electronically controlled loads, HVDC links and inverter-based generators, and (ii) Long-term voltage instability which involves slower acting equipment such as tap-changing transformers, thermostatically controlled loads, and generator current limiters [88]. This application targets the long-term voltage instability, which typically manifests as a progressive voltage drop in the system ultimately leading to the collapse of system. The long-term voltage instability has been a contributing factor of many blackouts occurred around the world and

early detection enables taking manual or automatic remedial action to prevent voltage collapse.

The phenomenon of long-term voltage instability is typically described using the P-V curve (or Q-V curve) at a given load bus as shown in Figure 5-21 (a). The loadability margin (LM) illustrated on the P-V curve shown in Figure 5-21 (a) is an easily understandable indicator of proximity to voltage instability. Pre-determined LM thresholds can be used to alert the system operators of impending voltage instabilities. Tracing of the P-V curve starting from a given operating point needs a technique such as continuation power flow (CPF) and requires significant computational effort to apply to a large power system.

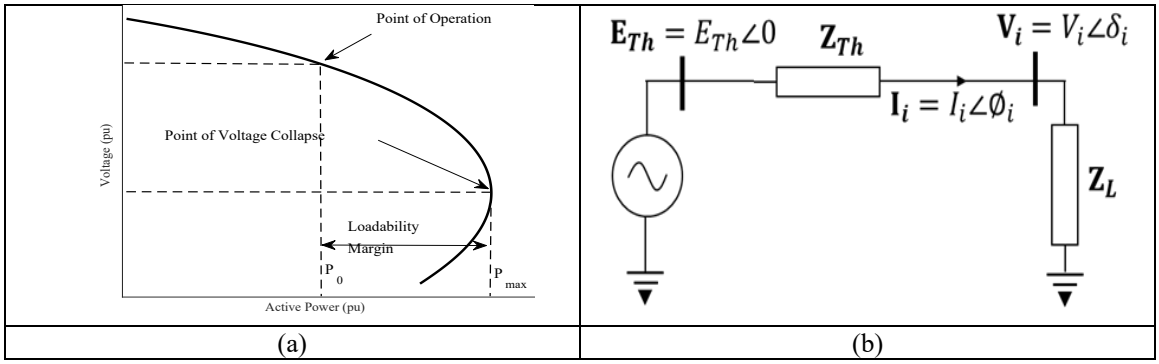


Figure 5-21: (a) P-V curve and Loadability Margin (b) Thévenin equivalent circuit of the network at load bus-i

5.2.2.2 Problem Statement

Voltage stability monitoring and prediction

The benefits of real-time voltage stability monitoring are well recognized. Although LM is a good indicator of proximity to voltage collapse, its computation using a technique such as CPF is time consuming for large systems due to iterative computations involved. Furthermore, accurate knowledge of the network topology, which can be undergoing changes during a disturbance, is required for CPF. Therefore, evaluation of LM using direct methods is not practical for real-time voltage stability monitoring.

Use of voltage stability indices (VSI) is an alternative method of assessing the long-term voltage stability. In general, VSIs are derived with respect to a load bus while considering the rest of the system as an equivalent circuit as shown in Figure 5-21 (b). Most VSIs are founded on the maximum power transfer theorem but differ in focus (load bus or transfer corridor), extent of information used (local or wide-area measurements, requirement of topology and system admittance data), simplifying assumptions, etc. There are some VSIs based on other considerations such as reactive power reserves, power losses in a region of interest, etc. A number of these VSIs can be computed using real-time measurements with a computational effort feasible for real-time implementation. However, when the topology and the operating conditions change (especially during an evolving abnormal situation), the accuracy of different VSIs is affected depending on the underlying principle. Therefore, relying on a single VSI to detect impending voltage instabilities has drawbacks.

There is a potential for improving the accuracy of voltage stability assessment by combining several VSIs, particularly those based on different principles. Furthermore, an easily understandable indicator such as LM would be more useful for system operators in taking remedial actions. Therefore, the approach proposed here explores the possibility of using multiple VSIs to predict the loadability margin. However, the relationship between the LM and VSIs is not straightforward and therefore a machine learning type approach is required to learn the implicit relationship between the LM and the VSIs.

5.2.2.3 Real-time Voltage Stability Monitoring System

This scheme consists of three subsystems: phasor measurement unit (PMU) infrastructure, real-time monitoring system at the control center and the offline mashing learning model (MLM) training system as shown in Figure 5-22. The PMU infrastructure provides real-time synchrophasor measurements from selected locations of the power system. The voltage stability monitoring (VSM) system collects PMU data and predicts the LM of the power system using a set of MLMs trained offline. PMU measurements are validated and conditioned to remove bad data, and then used to compute several VSIs. MLMs use computed VSIs and phasor data for predicting the LM. Predictions of the ensemble of MLMs are aggregated to obtain the final prediction, which will be displayed to the operator through a voltage monitoring dashboard.

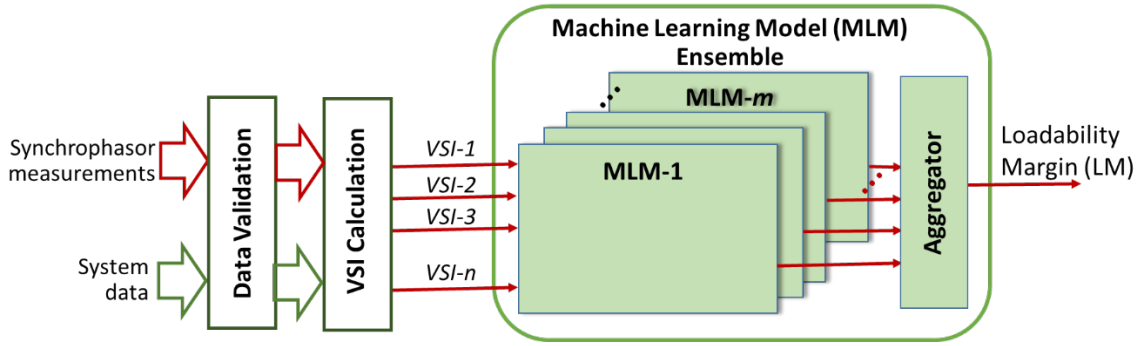


Figure 5-22: Proposed real-time VSM system

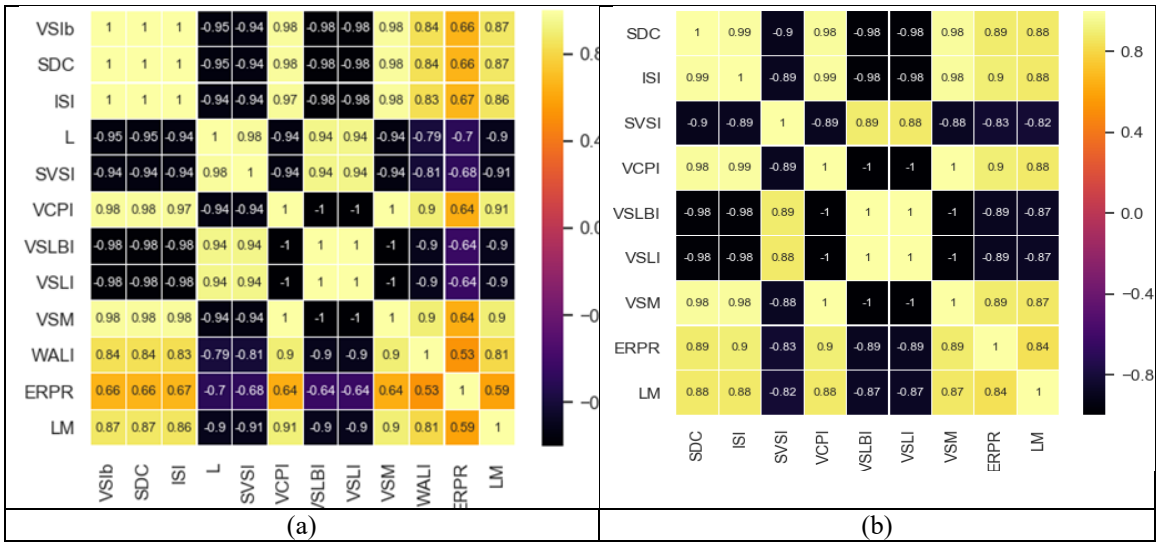
5.2.2.4 Feature Selection and Offline Training Process

The first step of the offline MLM training process is to generate a large sample of potential operating conditions of the power system, including under different contingencies. These operating conditions are then fed to a CPF program to calculate the bus voltages and LMs as the system load increases. Then these data are used to calculate the set of candidates VSIs listed in Table 5-2, which are selected based on the suitability for real-time implementation. A large database is created by computing VSIs and the corresponding LMs at different operating points obtained by sampling the CPF results.

Table 5-2: Candidate VSIs for MLMs (see [89] for VSI definitions/references)

Name of VSI	Abbreviation
Voltage Stability Load Index	VSLI
Voltage Collapse Proximity Index	VCPI
Voltage Stability Load Bus Index	VSLBI
S Difference Criterion	SDC
Simplified Voltage Stability Index	SVSI
Impedance matching Stability Index	ISI
Voltage Stability Margin	VSM
Reactive Power Reserve Index	ERPR
L-index	L
Wide-Area Loss Index	WALI

The second step of the development is the feature selection for MLMs. The correlation between each VSI and LM is analyzed using the Spearman's rank correlation coefficients to select the most suitable VSIs to be used as input features for MLMs. This process can also eliminate the VSIs that provide redundant information. Examples of this analysis performed for IEEE 14 and 118 bus test system are presented in Figure 5-23.



(a) (b)

Figure 5-23: Correlation heat map of candidate VSIs and LM for (a) IEEE 14 bus system and (b) IEEE 118 bus system

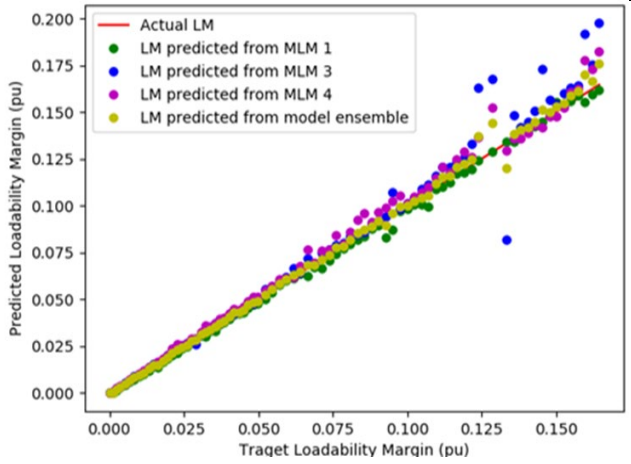
The third step is the offline training and validation of machine learning based LM prediction system. The most important features (VSIs) selected in step two are used to generate the input-output data pairs to be used in training and testing of MLMs. Each data set includes a set of input VSIs computed at a particular operating point and the LM at that operating point. These data sets are structured in the form of [VSI_1, VSI_2, …..VSI_n: LM]. The data is divided into training and testing data (roughly in 3:1 proportion). Several MLMs are designed considering other criteria such as those VSIs computed using only the

measurements at a local bus. Also, one MLM is designed to directly use the voltage phasors as inputs and the data for it are structured in the form of [$|V_1|$, δ_1, …..$|V_n|$, δ_n : LM]. The locations of voltage measurements are selected based on a recursive feature elimination method. Descriptions of MLMs are given in Table 5-3.

Table 5-3: Descriptions of different MLMs in the ensemble

MLM	Inputs
MLM 1	Voltage magnitudes/phase angles of selected buses
MLM 2	All of the candidate VSIs
MLM 3	VSIs which are calculated using only the local measurements of the considered bus
MLM 4	VSIs with high correlation (correlation > 0.9) with LM

Random Forest (RF) regression machine learning algorithms is selected as the most feasible algorithm among the other candidates through preliminary studies. After training each MLM, the accuracy of the MLMs were tested and the less accurate MLMs were discarded, for example MLM 2 in Table 5-3 is eliminated. Finally, a set of weights is determined to aggregate the outputs from MLMs in the ensemble using weighted average method. Typical prediction results for testing data, using the individual MLMs and the MLM ensemble are compared in Figure 5-24 (a). When the prediction results are 100% accurate, they would coincide with the red diagonal line. The predictions made with MLM 3, which uses only local measurements deviate significantly from the actual values, while the predictions made with MLM 1, which directly uses voltage magnitudes and angles are quite accurate. However, the most accurate predictions are obtained with the ensemble that combines the outputs of all MLMs. The accuracy of predictions (RMS Error (RMSE) and R^2 value) obtained with different weightages in combining MLM outputs is shown in Figure 5-24 (b). The 2:1:1 combination of MLMs 1, 3, and 4 is considered for the final implementation, and it gives an R^2 value of over 0.998 for the IEEE 14 bus system and an R^2 value over 0.975 for the IEEE 118 bus system.



(a)

Test System	Weightage ratio MLM1: MLM3: MLM4	RMSE	R^2
IEEE 14	1:1:1	0.0057	0.9983
	2:1:1	0.0053	0.9985
	1:2:1	0.0061	0.9981
	1:1:2	0.0062	0.9980
	2:2:1	0.0056	0.9984
	2:1:2	0.0056	0.9983
	2:1:2	0.0062	0.9980
IEEE 118	1:1:1	2.8252	0.9725
	2:1:1	2.6935	0.9752
	1:2:1	2.8551	0.9717
	1:1:2	3.0425	0.9663
	2:2:1	2.7216	0.9748
	2:1:2	2.8537	0.9713
	2:1:2	2.9715	0.9686

(b)

Figure 5-24: (a)LM prediction accuracy comparison for IEEE 118 bus test system, (b) Effect of weightages on the accuracy of ensemble output

5.2.2.5 Real-Time Implementation and Testing

The proposed real-time VSM system was implemented and tested using experimental setup shown in Figure 5-25, which consists of an RTDS® digital real-time digital simulator and PhasorSmart software platform. The local area network implemented using a RuggedCom™ RSG2288 utility grade Ethernet switch facilitates the communication of the synchrophasor measurements from the PMUs simulated in RTDS to a PC that runs the software platform. The synchrophasor application program consists of two main platforms: ePDC virtual Phasor Data Concentrator (PDC) and Synchronous Data Server (SDS), which are responsible for concentrating the PMU measurements from different PMUs. SDS publishes data to the VSM application program developed using C++. This program implements the VSI calculation and LM estimation using the trained MLMs. A monitoring dashboard is implemented using web based Grafana visualization tool to provides a user-friendly monitoring interface as shown in Figure 5-26 (a) [88]. A sample of LM predictions using trained MLMs during various system events is shown in Figure 5-26 (b).

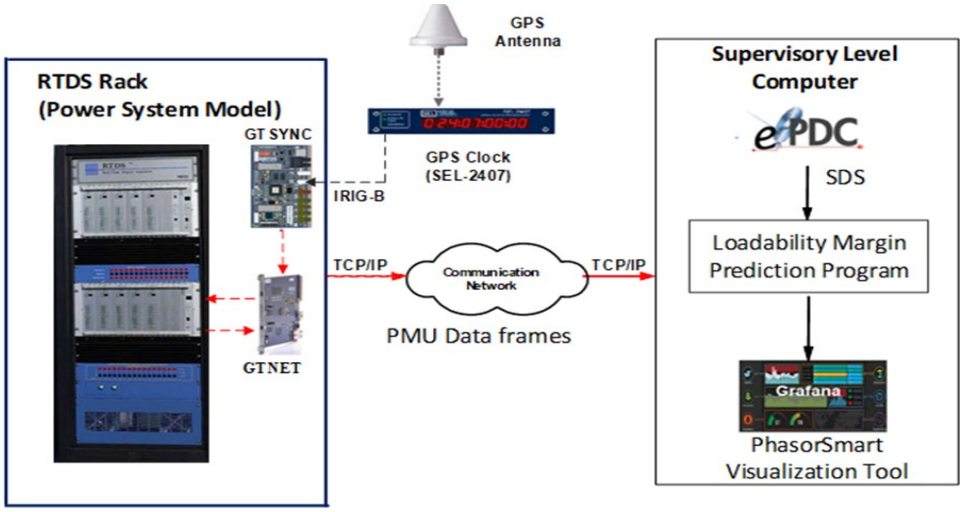
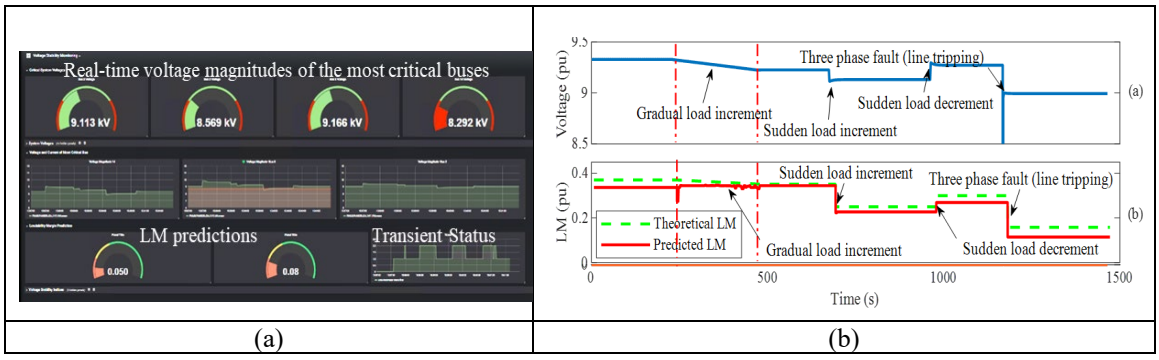


Figure 5-25: Experimental Setup



(a) (b)

Figure 5-26: Examples of (a) voltage stability monitoring dashboard, and (b) variation of real-time predicted LM with system changes/events for IEEE 14 bus test system

5.2.3 Detecting Relay Misoperations Using Field Data

Although relay non-operations are detected in real time and supported by local and remote backup protection, their incorrect operations, or misoperations, are never detected in real time. In a survey of four utilities conducted by PSRC, yearly misoperations were found to be 15.4%, 9.5%, 52.6%, and 28.6% at 345 kV voltage levels [90]. A study of hidden failures [91] shows these can be responsible for unintended operation of distance relays, breaker failure relays, underfrequency relays, and undervoltage relays. Blackout reports show that misoperations have contributed to either initiating or exacerbating the progression of large-scale instabilities that take anywhere between a few minutes to a few hours to result in blackouts [92]. Load encroachment, a common misoperation of distance relays, has occurred during the majority of documented blackouts.

Since synchrophasor measurements capture signatures of dynamic events in power systems, a disturbance identifier tool can be used to flag off any relay misoperation. The concept, called supervisory protection and dynamic event visualization, is shown in Figure 5-27. As seen from the figure, the application would reside inside a PDC and classify disturbances in real time. If a relay operates, but a fault is *not* detected by the classifier, a flag is set off, indicating possible misoperation. If, on the other hand, classifier also detects a fault, the relay operation is validated. This algorithm is not used to classify misoperations due to slow operation or failure to operate when a fault is present.

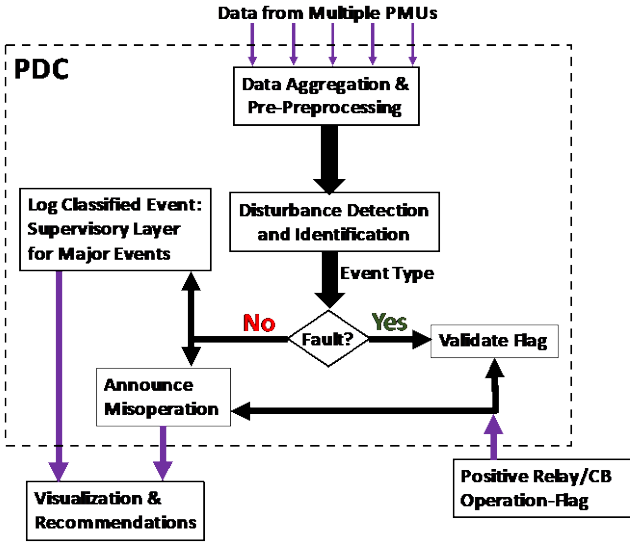


Figure 5-27: Supervisory protection and dynamic event visualization.

There are many classifiers proposed to identify dynamic events using PMU data, but few describe the exercise using field data. This section describes an effort made using field data. It will underscore the obstacles faced in creating a classifier when field data are used. It will also summarize how these obstacles can be tackled. A comprehensive description of the material presented here can be found in [93].

The data available to the researchers in this case consisted of all the disturbance files captured by four PMUs owned by a utility in the Southwestern USA from 2007 to 2010. Data consisted of the positive sequence voltages, frequency, and rate of change of frequency measured in the 345 kV transmission network at 30 fps reporting rate. After discarding files which suffered significantly from data loss, and files that were simply continuation of an event-recording, a total of 1013 disturbance event files were selected for this study. These are not enough to train and test any classifiers, which constituted the first hurdle. Additionally, the utility log captured only 84 of these 1013 events - 23 generation loss events, 58 faults, and 3 line-trip events. Thus, majority of events recorded by PMUs were unknown. Among them, many events might have occurred outside the utility's boundaries (i.e., in neighboring utilities). This situation creates two additional hurdles: 1) the exact number of classes, which in this case would be types of disturbance events, are unknown, and 2) ground truths required to test a classifier are inadequate.

To overcome these hurdles, the following strategy was adopted:

- Since the number of classes that would encompass all PMU disturbance data are unknown, the first step is to know how many clusters these data can be segregated into. One cluster would ideally correspond to one type of disturbance events and consists of a group of disturbance files recorded for all occurrences of this type of events.
- After obtaining the clusters, each cluster will be mapped to a disturbance type. Ideally ground truths would be used for this step. However, since the ground truths are available for only three events, domain expertise was used.
- A pattern-creation technique and a classifier need to be chosen for precise classification.
- Since the field data are inadequate to train and test a classifier, additional data will be generated through simulation of the utility's network. Preferably, field data will be used for testing the chosen classifier.

An unsupervised clustering method – agglomerative hierarchical clustering technique called AGNES (AGglomerative NESting) - was chosen to find clusters contained within the 1013 disturbance files [94]. A previously published method for pattern creation – Minimum Volume Enclosing Ellipsoid, or MVEE – was used to create and feed patterns to the clustering technique. This method uses domain transform on *all* data-streams coming to the PDC to create a multi-dimensional ellipsoid. The geometrical properties of the ellipsoid are then taken as patterns. This is a computationally intensive approach which was improved in the later part of our study. The results from the clustering technique are shown in Table 5-4. Nineteen clusters emerged. Some of the clusters were identified straight away as they contained the known ground truths. However, 10 clusters remained unidentified. The good news was that no two ground truths fell into the same cluster, which provided credibility to this approach.

In order to identify events falling in the unknown clusters, domain knowledge was used. The unknown clusters were identified as shown in Table 5-5. Here, LS stands for load switching, and SCS stands for shunt capacitor switching. Pumping units were synchronous motors in the utility system. To further test the veracity of the disturbance types determined by domain knowledge, all the disturbances were simulated on the PSLF® software, using

the WECC model of the utility system available in the software. Twenty instances of each disturbance were simulated and then all the event files from simulated and field data were grouped together and re-classified using the same method. The same 19 clusters emerged and there was no cross-classification. This provided credibility to the process of discovering groups hidden in the PMU dataset.

Table 5-4: Clusters formed by using agglomerative hierarchical clustering approach.

Group Name	Total Events	Known Events	Type of Disturbance
Group-4	70	3	*Generation Loss*
Group-11	58	7	
Group-14	59	1	
Group-15	28	12	
Group-8	40	21	*Fault*
Group-12	74	19	
Group-13	16	11	
Group-16	9	7	
Group-7	36	3	*Line Trip*
Group-1	147	-	*Unknown*
Group-2	85	-	
Group-3	93	-	
Group-5	29	-	
Group-6	44	-	
Group-9	64	-	
Group-10	80	-	
Group-17	13	-	
Group-18	47	-	
Group-19	21	-	

Table 5-5: Disturbance events identified for initially unknown clusters.

Group Name	Total Events	Tested Events	Type of Disturbance
Group-5	29	20	*LS - On*
Group-6	44	20	*LS - Off*
Group-17	13	20	*SCS - On*
Group-19	21	20	*SCS - Off*
Group-3	93	20	*Pumping Unit Loss*
Group-18	47	20	
Group-1	147	20	*Generation Loss*
Group-2	85	20	
Group-9	64	20	*Remote Fault*
Group-10	80	20	

Once the disturbances were discovered, the next step was to build a classifier. However, due to too few field data, more simulated data had to be generated for all types of disturbances. Table 5-6 shows the simulated data generated for each of the seven major events (line-trip was excluded).

Table 5-6: Number of simulated events for the seven disturbance types under study.

class	events
Fault (FLT)	1260
Generation loss (GL)	558
Load switching (LS) off	348
Load switching (LS) on	354
Reactive power switched out	497
Reactive power switched in	442
Synchronous motor switching off	36
Total	3495

Since the disturbance traces were rich with time-domain signatures, a time-domain pattern creating method called *shapelets* was adopted to extract the patterns. This method was implemented in our work as domain-specific shapelets (Dshapelets) [95] to save the computational efforts involved with domain transforms used by the MVEE method. *Slope Sequence of Shaplets (S^3)* was a pattern derived from the shapelets. A slope formed from raw data (S^2) was also chosen as a pattern. Other more popular patterns involving domain transformation – Discrete Fourier Transform (DFT), Discrete Wavelet Transform (DWT), fast variant of discrete S transform (FDST), and Principal Component Analysis (PCA) – were also chosen for a comparative study. Finally, some statistical metrics formed from time-series data were also chosen as a pattern. The measurement noise in the field

waveform [96] was kept as it was. Additionally, since data streams from all PMUs would be affected in the same way by a given disturbance, the data stream with the strongest signature was identified using an energy-based index, and used for the classification, thus reducing the computation burden on the classifier [95].

Table 5-7: Training (3495 simulated events from 7 disturbance types), testing (81 actual events with FLT and GL)

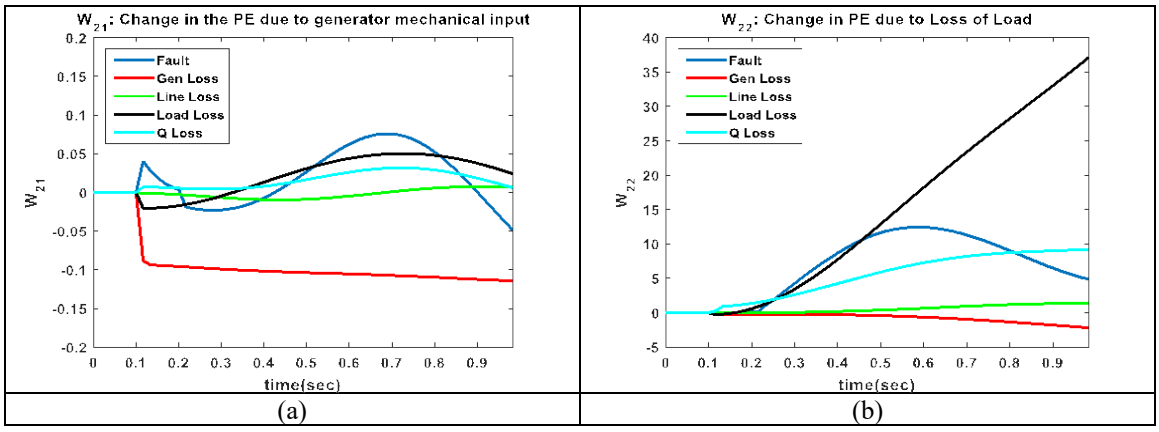
	Voltage		Voltage and Frequency	
	1NN	SVM	1NN	SVM
Raw data	49.4	81.5	61.7	71.6
DFT	55.6	71.6	72.8	71.6
DWT	50.6	75.3	60.5	71.6
FDST	**86.4**	69.1	**88.9**	74.1
PCA	71.6	71.6	71.6	71.6
S^2	50.6	**87.7**	56.8	85.2
Dshapelet	56.9	74.1	61.7	71.6
S^3	77.8	82.7	72.8	**100**
Statistics	32.1	81.5	54.3	84.0

Support vector machines (SVM) and K – Nearest Neighbor (1NN when K=1) were chosen as classifiers, as they had shown the best performance in earlier studies [92]. Table 5-7 shows the results from training the classifiers for all seven disturbances with simulated data and testing it with all the field data. *Slope Sequence of Shaplets (S^3)* is seen to outperform other patterns. It is observed that voltage and frequency both need to be used as inputs, since each has unique signatures for disturbances [95][6]. Also observe that the raw data based pattern (S^2) performed better than the methods requiring change of domains. This underscores the fact that there is no one "magic" pattern that works for all applications.

Table 5-8: Training (90% of all 3495 simulated events with 7 disturbance types), testing (10% of all 3495 simulated events with 7 classes); ten-fold cross validation.

	Voltage		Voltage and Frequency	
	1NN	SVM	1NN	SVM
Raw data	80.6	75.2	86.9	77.8
DFT	74.8	66.5	81.8	64.3
DWT	80.5	74.1	86.9	74.9
FDST	73.9	70	82.9	77.4
PCA	36.7	36.1	36.1	36.1
S^2	**87.5**	**80.5**	**93.3**	**91.2**
Dshapelet	78.2	69.9	86.3	79.6
S^3	79.8	71.7	92.5	88.5
Statistics	77.1	69.2	82.8	76.2

Now, the classifier was tested for all seven events using only simulated data due to unavailability of field data for all seven disturbances. The results are shown in Table 5-8. Although both S^3 and S^2 patterns give the best performance, the accuracy drops. Further investigation revealed that the only confusion was between Generation Loss and Load Switched On, since they impact voltage and frequency in the same way from a physical viewpoint, so the signatures are not distinguishable. This is a drawback of the data-driven method. To overcome this drawback, a physics-based method based on the system energy function was designed [93]. In this method, *components* of the energy function were monitored to track each event. Figure 5-28 (b) shows the component that captures the change in energy due to loss of load and Figure 5-28 (a) shows the component that captures change in energy in generator mechanical input [93]. The two events are distinctly different. This exercise underscores the importance of an approach of physics-informed machine learning, whenever feasible.



(a) (b)

Figure 5-28: Energy change pattern due to (a) loss of generator mechanical input, and (b) loss of load

Finally, it was observed that faults had such distinct signatures that they were identified with 100% success. Time taken to determine whether an event is a fault or not was less than 0.4 ms. Even after adding sensing, PMU measurement and communication delay, which are reasonably estimated between 50-130 ms according to the IEEE C37.118.2-2011 standard, the supervisory protection can be performed in real-time, because the time taken for misoperations to cause large scale instabilities has been observed to be in minutes to hours. Most of the misoperations do not even result in instabilities.

5.2.4 Post-Event Analysis

In digital substations, IEDs with event recording capabilities are used for the protection and control functions of the power equipment. Some of the substations also have Digital Fault Recorders (DFRs) that capture event data for post-event analysis. When a fault or an event happens in the power grid, the IEDs and DFRs can perform the pre-fault and after-fault data recording of circuit statuses and electric parameters by event triggers. A cascading fault event often involves a large area and many IEDs. To manually analyze the root cause of the fault requires a lot of engineering effort. The post-event analysis can utilize AI/ML to help with creation and analysis of fault trees using the protection and control logic in the IEDs [97]. Each primary equipment (e.g., transformer, line, generator.) has a fault tree linked to the IEDs that are used to perform the protection and control. It starts by collecting the event COMTRADE files from the IEDs and finding the matched fault trees in the library such that the picked up elements will be searched through the tree and highlighted in the fault tree together with a time stamp. For a cascading fault event that involves multiple IEDs, the tool will highlight all the fault trees involved and create a combined COMTRADE file with all the elements that have picked up.

A Finite State Machine (FSM) model of the post-event analysis and visualization is shown in Figure 5-29. When started, the system is continuously monitoring the event trigger. The monitoring time window is user configurable and any other triggers received within this window of the aforementioned trigger will be considered as related events and similarly processed. The monitoring time window is also a wait-time for “Event/COMTRADE Collect” state. All the IEDs that have new Event or COMTRADE files will be automatically connected for event file downloading and the retrieved files will be stored in a specific folder for analysis.

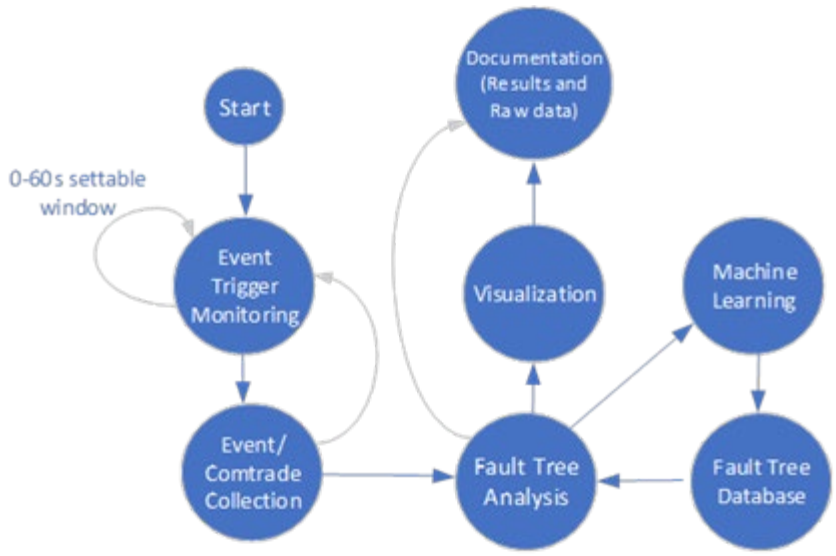


Figure 5-29: FSM Model of the Post-Event Analysis

The application takes the new event files and processes them by matching the fault trees in the database. Both the IED name and event tag will be used for the searching criteria. Once the event tag is found in the fault tree, it will then be highlighted in the fault tree and displayed with the event pickup time. Figure 5-30 shows an example of traversed fault tree after a transformer bank tripped. The user can document the instantiated fault tree and raw data for later reporting.

In the case of an event tag which was picked up but is not included in the fault tree, the user can choose to add it to the fault tree or to the ignoring list. Machine learning can be implemented in the tool with some built-in operation matrix for each type of relay. The ignoring list contains the list of event tags that may be duplicated or not related to the root cause of fault, thus no need to include in the fault trees.

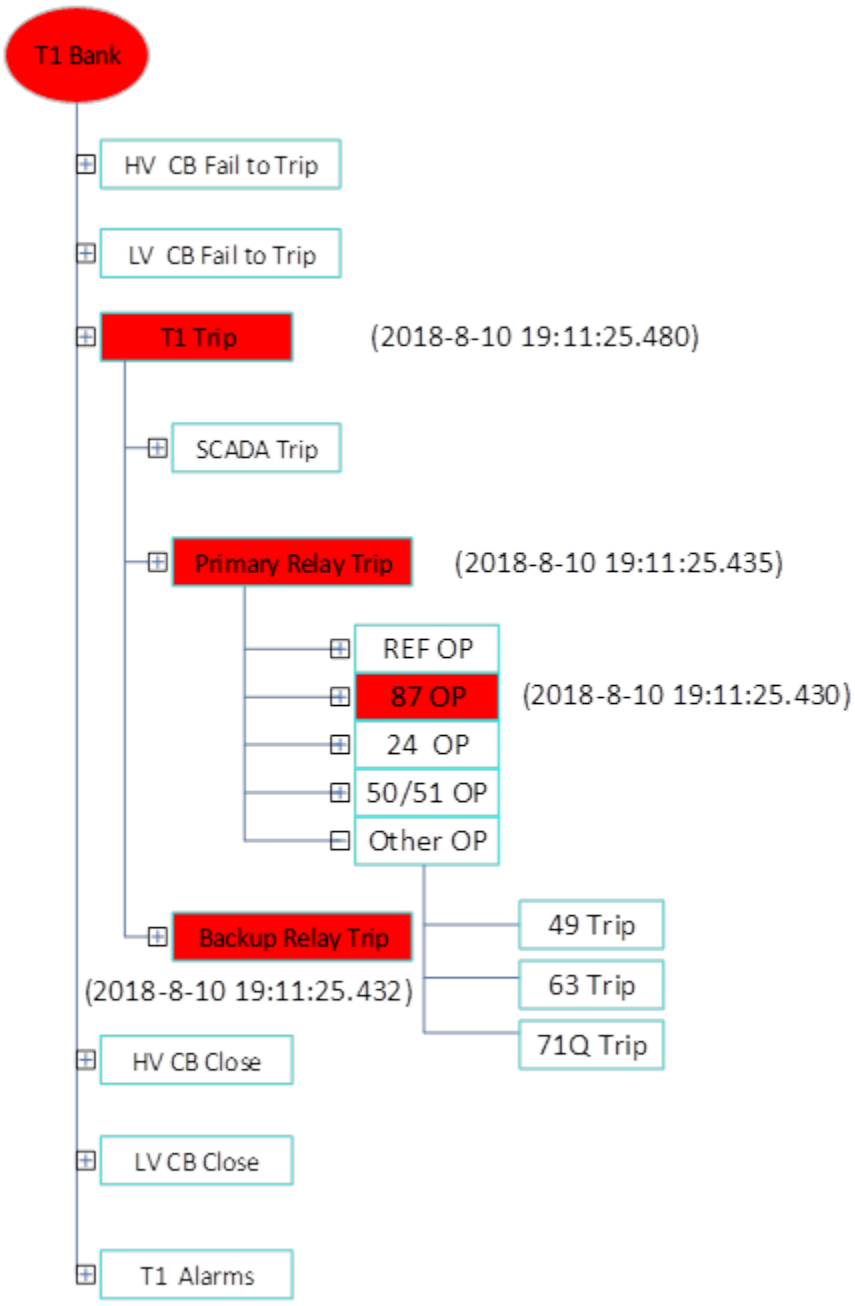


Figure 5-30: Fault Tree Construction for Transformer Bank X

5.2.5 Application of AI/ML in Travelling Wave Protection

This section provides application examples of ML for traveling wave (TW) protection of AC and DC systems. In the following, first, an ML-based algorithm for the TW protection of AC distribution systems is discussed. Then, ML application for the TW protection of DC systems is elaborated. While TW protection is well developed for AC transmission

lines and HVDC lines, it is significantly more difficult to apply to distribution systems with many taps, transformers, and customers. For this application, AI/ML is used to learn the expected high-frequency fault signatures from multiple reflections in networks to determine the fault type and location in less than 1 ms.

5.2.5.1 TW Protection Background

TWs are electromagnetic waves propagating along the power system equipment such as lines or cables when a disturbance (e.g., fault, lightning, switching, etc.) occurs. When the high-frequency TWs reach a new environment with different circuit parameters (e.g., at a line terminal), a portion of the incident TW is reflected while the other portion is refracted to the neighboring equipment. Depending on the circuit parameters at both sides of the terminal, the amplitude of reflected and refracted TWs changes accordingly. The TWs are also reflected and refracted again after they reach the fault location or line terminals. One of the common approaches to extract high-frequency TWs is to utilize Discrete Wavelet Transform (DWT). A DWT is any wavelet transform (WT) for which the wavelets are discretely sampled. The WT has been widely utilized as an effective tool for the simultaneous analysis of waveforms in time and frequency domains.

In order to effectively construct wavelets over a wide frequency range, multiresolution analysis (MRA) is a practical approach for fully implementing the DWT. MRA details the procedure to obtain an orthonormal wavelet basis with compact support. MRA can be implemented by a series of high-pass and low-pass filters and decimators as shown in Figure 5-31. As seen, the outputs of low and high-pass filters at each level are $a_i[\mathrm{n}]$ and $d_i[\mathrm{n}]$. The output of the low-pass filter at each level is passed through the next level for constructing wavelets for the next decomposition level. The low-pass filter outputs are referred to as scaling coefficients while the high-pass filter outputs are called wavelet coefficients. Assuming that the initial sampling frequency of WT is set as f_s, then the frequency range of each level is shown in Figure 5-31. The wavelet coefficients are of interest since they better represent the high-frequency behavior of TWs.

Once the MRA-based wavelet coefficients are identified, Parseval's theorem is used to calculate the energy corresponding to the identified coefficients. If the scaling function and mother wavelet form an orthonormal basis, then Parseval's theorem can be used to build a relationship between the calculated wavelet coefficients and the energy spectrum of the fault signal (i.e., cable's voltage or current measurement). Under this condition, Parseval's theorem states that the energy of the fault signal can be described mathematically in terms of the expansion coefficients (i.e., the integral or sum of the square of the original function is equal to the sum of the square of the coefficients). The DWT can split the energy of fault signals in time and frequency domains. With Parseval's theorem, one can effectively interpret the high-frequency signatures of TWs by relating the current or voltage of the TW energy to the energy of the wavelet coefficients. The fault signal can be described by the wavelet coefficients of different ranges.

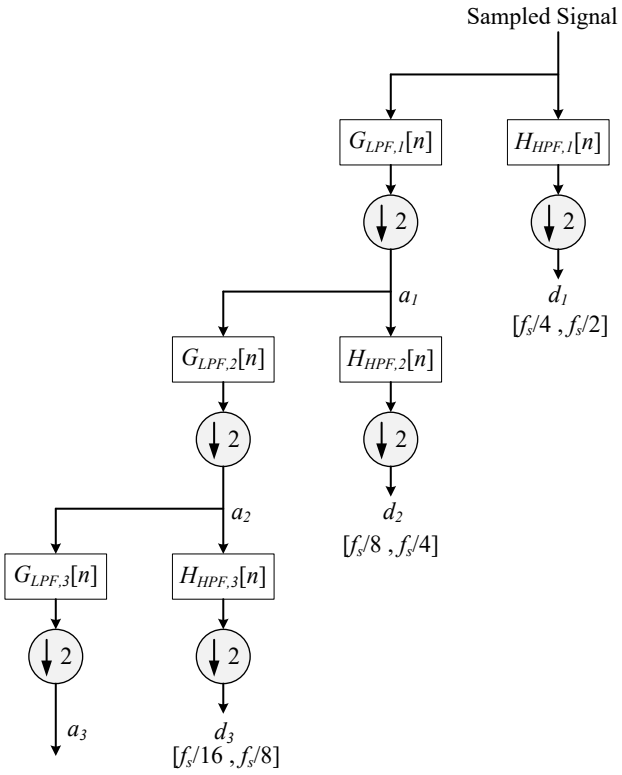


Figure 5-31: MRA block diagram.

5.2.5.2 TW Protection of AC Distribution Systems Using ML [98]:

The goal of this approach is to use TW measurements as inputs to a machine learning algorithm to detect and locate different types of faults in a distribution feeder.

ML Approach

The method can be separated into two main stages: the signal processing block that aims to process the measured signals (+/- 50 μs since TW time of arrival) and extract useful information, and the Machine Learning stage where several algorithms are trained to perform the actual fault location and classification. The workflow is shown in Figure 5-32.

Signal Processing: As it can be seen, the first part of the signal processing block consists of applying some transforms to both the 3-phase voltage and current signals. Here, the Clarke, Karrenbauer, and DQ0 transforms are employed. Numerical results showed that the larger number of features, the better performance. Each transform performs a different operation on the measured data, which is later used to create additional features to train the ML algorithms.

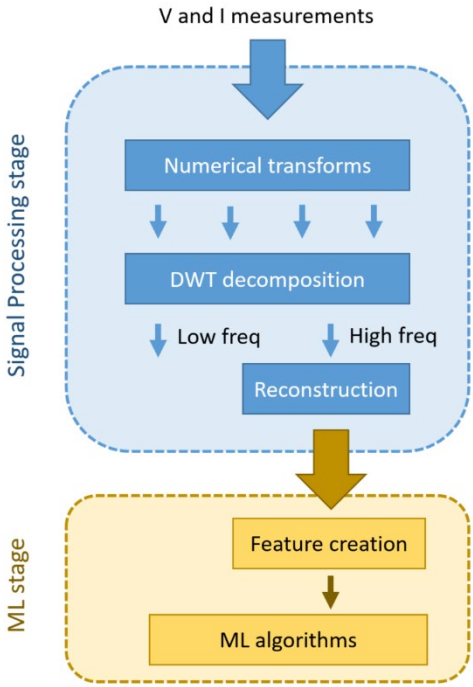


Figure 5-32: AI-based traveling wave fault location algorithm

Herein, a comparison between several algorithms has been made, with special emphasis on ensemble learning, which consists of combining multiple ML models in order to create a more accurate and powerful model. Currently, it is one of the most used techniques to improve the performance. There are several techniques to implement ensemble models, such as bagging, boosting, or stacking. For comparison, a Random Forest (RF) algorithm (example of bagging), implemented on scikit-learn library, has been selected, along with Google's TensorFlow Boosted Trees (BT) estimator (Gradient Tree Boosting). Some notions of these techniques will be given later. These particular algorithms have been selected because of their excellent accuracy, computational efficiency, and scalability for a larger number of samples/ larger system (which is where they outperform other common ML algorithms, such as Support Vector Machines (SVMs)). Finally, some of the state-of-the-art Machine Learning algorithms integrate different types of models into one ensemble model that takes advantage of the individual skills of each of the components (known as stacking). In this line, one model that groups the two aforementioned individual methods (RF and BT) has been implemented.

Simulation Test System: The system consists of one 12.47 kV variable-length distribution line that is connected to a load of 300 kVA per phase. Faults occur on the load bus. Simulations were performed in PSCAD for 160 fault locations (increasing the length in steps of 25 meters), for 3 fault types (single-line-to-ground, line-to-line, and 3-phase), and for 7 resistance values ranging from 0.01 to 10 Ω. This sums up to 3360 simulations. Three-

phase voltages and currents are measured on the secondary side of the transformer at a sampling frequency of 10 MHz. The system is shown in Figure 5-33.

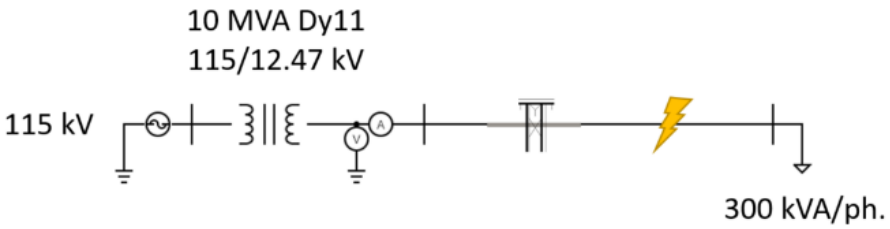


Figure 5-33: Schematic of the system

Fault Type Classification: For the fault type classification task, the algorithms are able to discern whether the measurements are of a Single-Phase fault, Double-Phase fault, or Three-Phase fault. Accuracy is defined as the number of correctly predicted fault types over the total number of faults. Table 5-9 below summarizes the individual results of both RF and BT algorithms. As it can be observed, accuracy is close to 100% in both of them. The reason for this excellent performance is the large energy magnitude variations among each type of fault, which makes this task easy.

Table 5-9: Fault type classification accuracy.

Classifier	Accuracy
Boosted Trees	99.84%
Random Forest	100%

Fault Location: The results for the Boosted Trees and Random Forest regressors, along with the Stacking method can be observed in Figure 5-34. The average and the standard deviation of the errors are shown in Table 5-10. Most of the faults are located with an error well below 200 m.

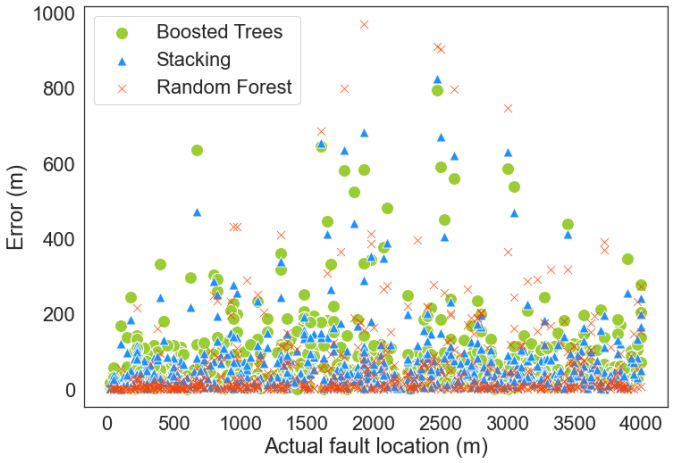


Figure 5-34: Fault location prediction errors relative to fault distance

Table 5-10: Mean and STD of prediction errors for fault location.

Regressor	Mean (m)	Standard Deviation (m)
Boosted Trees	72.64	99.14
Random Forest	53.64	113.01
Stacking	62.95	94.77

As it can be seen in Figure 5-34, the Random Forest model tends to have lower errors in most of the cases, but there are a few outliers. However, the Boosted Trees algorithm is much more consistent. The stacking of both algorithms leads to lower values and deviation in prediction errors.

5.2.5.3 TW Protection of DC Systems Using ML [99]:

In order to study the high-frequency fault signatures of DC systems, a simple DC system (shown in Figure 5-35) is modeled in PSCAD/EMTDC. This circuit includes a controllable DC voltage source, one cable with the length of 3000 m, and a DC load with a resistance of 10 Ω. The nominal voltage of this system is ±375 V. The cable is modeled using the frequency-dependent distributed parameter model available in PSCAD/EMTDC. The cable specifications are provided in Figure 5-36. It is assumed that each pole is buried 1 m deep. The core conductor resistivity is 2×10^{-8} Ωm while sheath resistivity is 30×10^{-8} Ωm.

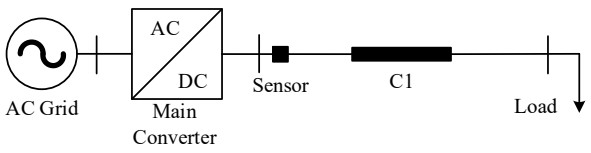


Figure 5-35 Simple DC microgrid system

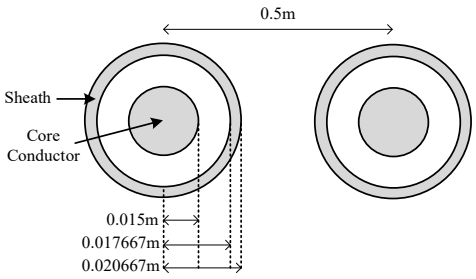
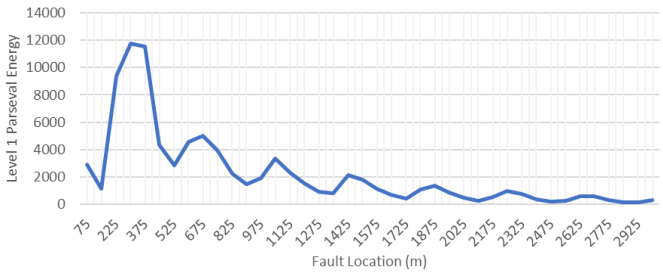
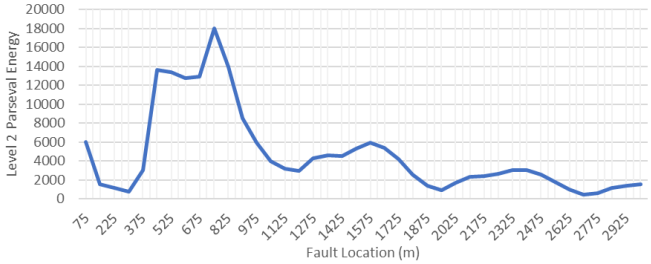


Figure 5-36: Cable configuration of the DC system in Figure 5-35

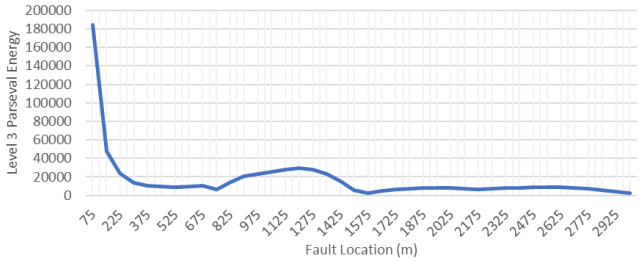
MRA is an effective tool to demonstrate the high-frequency fault signatures in different frequency ranges. In order to show how the Parseval energy of wavelet coefficients of each MRA's level is impacted for different fault locations, phase-to-phase faults are applied at every 75 m of cable C1 in Figure 5-35. First, it is assumed that DWT's sampling frequency is 1 MHz. The Parseval energy values for three levels of MRA applied to the current measured at the sensor in Figure 5-35 are shown in Figure 5-37. Herein, levels 1, 2, and 3 are associated with [250 kHz, 500 kHz], [125 kHz, 250 kHz], and [67.5 kHz, 125 kHz] frequency ranges. As seen in Figure 5-37 (a), the Parseval energy value is generally decreasing as the fault location gets closer to the end of the cable, however, some local peaks are observed that happen at every 375 m. A similar pattern is observed in Figure 5-37 (b); however, the local peaks occur at every 750-800 m. Finally, Figure 5-37 (c) shows that the local peaks occur at every 1500 m. As a rule of thumb, the number of local peaks approximately doubles from level 3 to level 2 and as well as from level 2 to level 1. In Figure 5-38, it is assumed that the DWT's sampling frequency is 2 MHz. Doing so, level 1 and 2 are associated with [500 kHz, 1 MHz] and [250 kHz, 500 kHz] frequency ranges, respectively. As seen in Figure 5-38, a similar pattern to Figure 5-37 can be observed. In general, with a higher DWT's sampling frequency, (i) more oscillations on the Parseval energy profile of fault currents are observed, and (ii) the first incident of TW can be detected faster. The latter is based on an inherent feature of TWs in which the higher frequency TWs travel faster with a lower magnitude.



(a)



(b)



(c)

Figure 5-37: Parseval energy values for different fault locations on the cable of DC system with 1 MHz sampling frequency in Figure 5-35: (a) MRA's level 1; (b) MRA's level 2; (c) MRA's level 3.

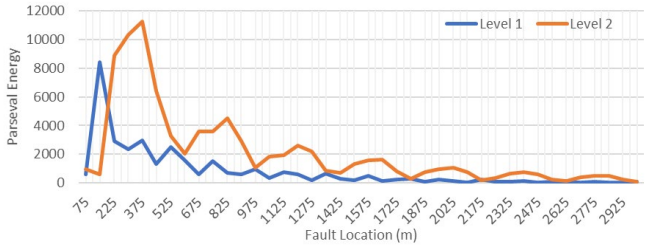


Figure 5-38: Parseval energy values for different fault locations on the cable of the DC system with 2 MHz sampling frequency.

The proposed fault classification and location algorithm is shown in Figure 5-39.

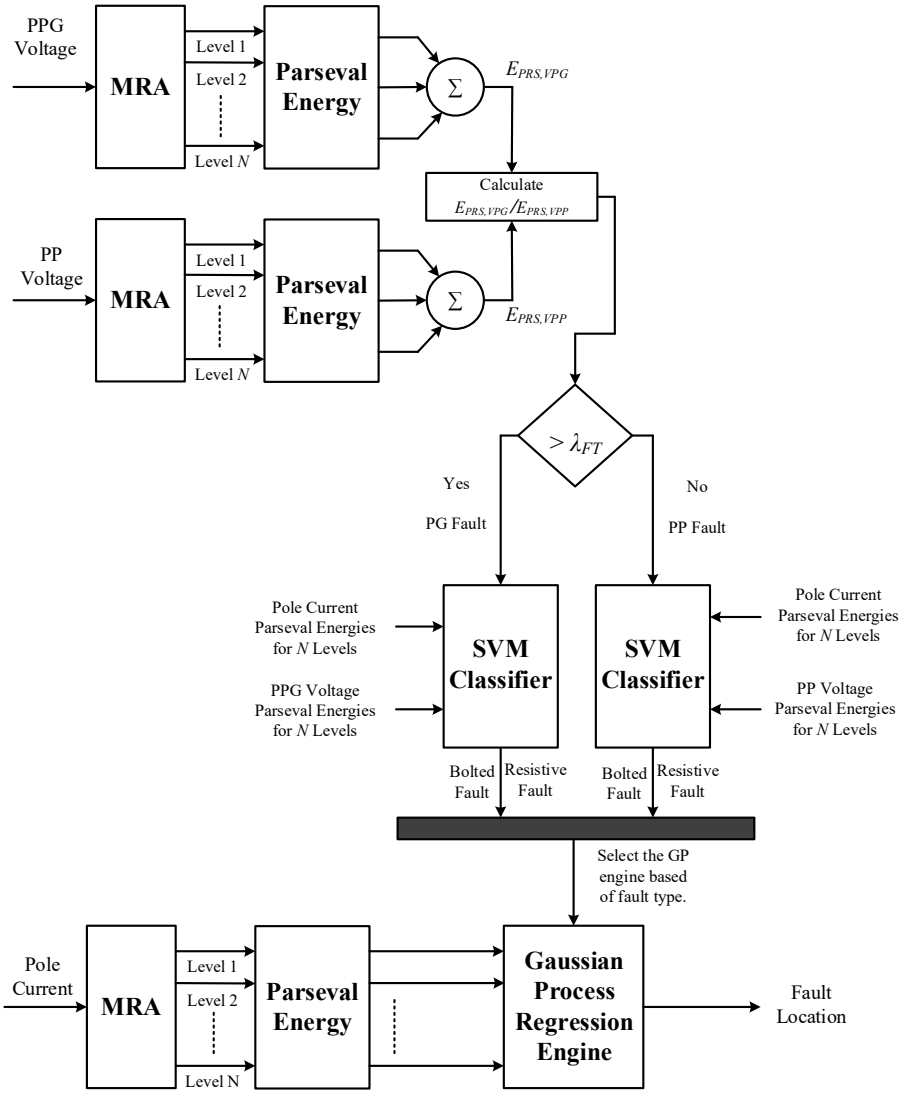


Figure 5-39: AI-based traveling wave fault classification and location algorithm for DC systems

Fault Classification: To distinguish between Pole-to-Pole (PP) and Pole-to-Ground (PG) faults, the Parseval energy of PP voltage at the sensor location is compared against the Parseval energies of the positive or negative pole to ground voltage. This procedure is shown in Figure 5-39. Simulation results show that for PG faults, the Parseval energies of positive or negative pole to ground voltage (PPG or NPG) are significantly higher than the Parseval energy of PP voltage. To this end, the algorithm first calculates the summation of the first N levels of Parseval energy values related to PP voltage and PPG voltage (i.e., $E_{PRS,VPP}$ and $E_{PRS,VPG}$ respectively). Then, $E_{PRS,VPP}$ and $E_{PRS,VPG}$ are compared against each other to determine the fault type. Herein, the ratio of $E_{PRS,VPG}/E_{PRS,VPP}$ is calculated and compared against λ_{FT} threshold. This threshold can be found by trial and error on the microgrid system. Since $E_{PRS,VPG}$ is significantly higher than $E_{PRS,VPP}$ for PG faults, λ_{FT} is

always greater than 1. As a rule of thumb, the threshold is selected at around 10% of the $E_{PRS,VPG}/E_{PRS,VPP}$ ratio for the remote end PP and PG faults.

Once the fault type (i.e., PP versus PG) is identified, a Support Vector Machine (SVM) classifier is used to determine if the fault is resistive or bolted. SVM has been proposed as a strong classification tool. For this purpose, multiple bolted and resistive faults are simulated at different locations of a cable (e.g., every 25 m) in a simulation software package (e.g., PSCAD/EMTDC). It should be noted that the fault resistance values adopted in the simulations depend on the DC microgrid conditions like voltage level or geographical location. After the simulation results are gathered, the SVM classifier is trained using the labeled Parseval energy values. For PP faults, the inputs to the SVM classifier are the N level Parseval energy values of pole current and PP voltage at the sensor location. For PG faults, the inputs to the SVM classifier are the N level Parseval energy values of pole current and PPG voltage at the sensor location. The output of the classifier is the fault resistance value.

Fault Location: Once the fault type is classified, first, the proposed approach determines if the fault is located on the primary cable. To this end, the Parseval energy value of the current flowing through the protection relay sensor, $E_{PRS,I}$, is calculated. Depending on the fault type, $E_{PRS,I}$ is compared against the precalculated Parseval energy value related to the remote end bolted PP, resistive PP, bolted PG, or resistive PG fault. The proposed fault location algorithm relies on the Parseval energy values gathered from MRA. The algorithm utilizes the first N levels of MRA, calculates the Parseval energy of the first TW incidents, and then utilizes Gaussian Process (GP) regression engines to find the fault location.

Simulation Results: The DC microgrid test system is illustrated in Figure 5-40. This DC microgrid system is based on a real DC microgrid system in the city of Albuquerque, NM. We have utilized high frequency (in the order of 10 MHz) field measurements from the actual microgrid to calibrate the created model in PSCAD/EMTDC. The DC microgrid is supplying four residential houses. Each residential house is described as a nanogrid (NG). Each NG included the residential house load, a PV system, a Battery Energy Storage System (BESS), and DC-DC converters to integrate NG into the rest of the microgrid. The PV system is associated with a maximum power tracking scheme. The size of the PV system in each NG is 10 kW while the size of BESS is 6 kW/12 kWh. The load of NG1, NG3, and NG4 is 56.25 kW and the load of NG2 is equal to 50 kW. It is assumed that the microgrid includes a community BESS and PV system with the size of 20 kW/40 kWh and 18 kW, respectively. The microgrid's main converter is modeled as a multi-level voltage source converter. The microgrid's grounding happens at the middle point of the DC link of the microgrid's AC/DC converter. The size of this converter is 500 kW. Load L1's size is 10 Ω. RC1 is 25 mΩ and LC1 is 20 μH. In Figure 5-40, relays R1 to R6 identify the location of the proposed fault detection and location algorithms. The cables configuration is provided in Figure 5-36. It is assumed that each cable has its own local protection.

To verify the effectiveness of the proposed scheme, bolted and resistive PP and PG faults are applied at different locations of all six cables in the DC microgrid test system shown in Figure 5-40. The fault resistance for both PP and PG faults is equal to 5 Ω. The length of each cable and its simulated fault locations are summarized in Table 5-11. In these

simulations, DWT's sampling frequency is 8 MHz. Six levels of MRA are used for the fault classification and location algorithm. For cables C1 and C6, 40 different fault locations are simulated; for cables C2 and C5, 30 different fault locations are simulated; for cable C3, 38 different fault locations are simulated; for cable C4, 22 different fault locations are simulated.

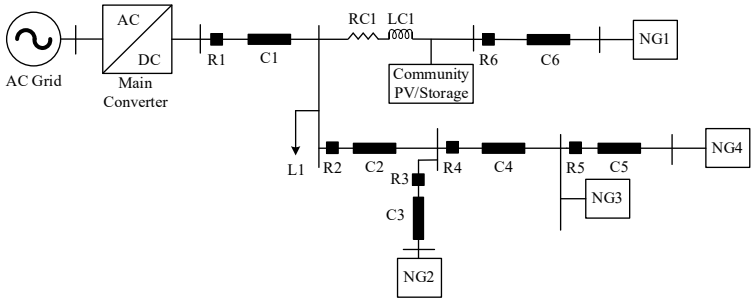


Figure 5-40: DC microgrid diagram

Table 5-11: Cable lengths and fault locations

Cable	C1	C2	C3	C4	C5	C6
Length	2000 m	800 m	1000 m	600 m	800 m	2000 m
Fault locations	At every 50 m	At every 25 m	At every 25 m	At every 25 m	At every 25 m	At every 50 m

The algorithm was able to effectively distinguish PP faults from PG ones by selecting 10 as the λ_{FT} threshold in Figure 5-39. To verify the performance of the SVM classifier for distinguishing bolted faults from resistive ones, two separate datasets were used for training and testing. 60% of the available data were used for training and 40% of them were used for testing randomly. In the SVM, a linear Kernel is used while the penalty factor for misclassified data is set to 1. The verification results rendered 100% precision in classifying bolted faults versus resistive faults using the six levels of current and voltage Parseval energies at cable C1. In Table 5-12, the fault location estimation errors using GP engines for all cables and different types of faults are summarized. The estimation error percentage is equal to the mean absolute error of the testing dataset over the length of the cable. For each cable, around 65% of the gathered datasets are used for training and the rest are used for testing. The training and testing datasets are selected randomly. The fault location estimation errors in Table 5-12 verify the effectiveness of the proposed fault location algorithm. In Table 5-12, the estimation error depends on the number of datasets available for training and the length of the cable. In general, simulating faults at more locations can increase the number of training datasets which in turn improves the performance of the GP regression engine in estimating fault location. The regression results for Cable C4 are illustrated in Figure 5-41. As seen, for both resistive and bolted PP and PG faults, the GP regression engine can effectively locate faults with small estimation errors. In all cases, the proposed algorithm is able to find the fault location in 200 μs.

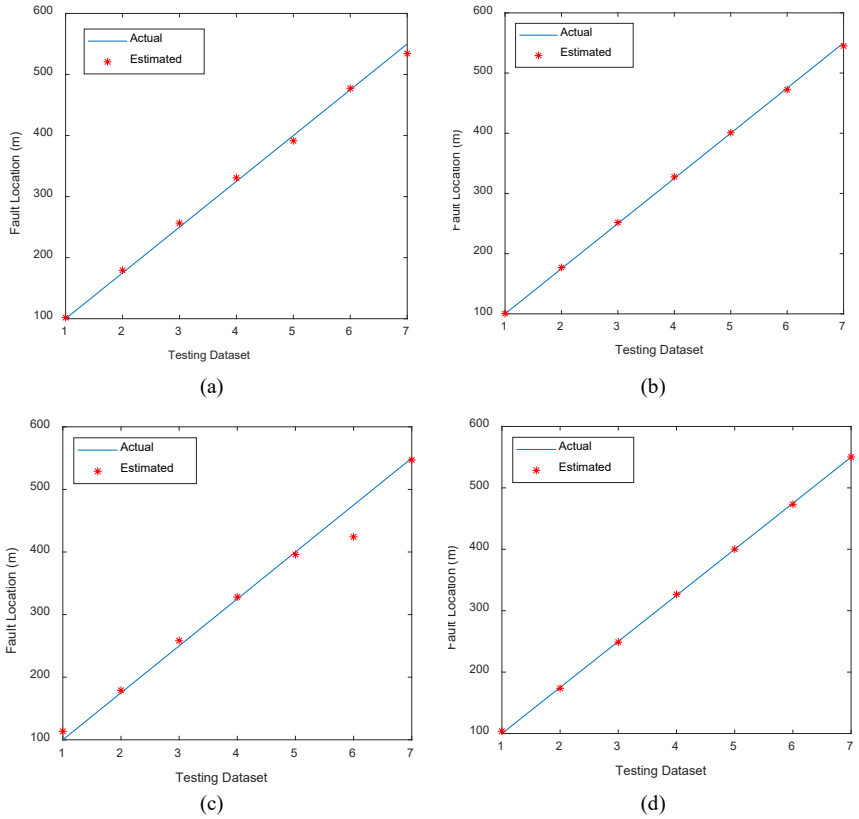


Figure 5-41: Regression verification plots for C4: (a) bolted PP faults; (b) resistive PP faults; (c) bolted PG faults; (b) resistive PG faults

Table 5-12: Fault location estimation error for DC Microgrid 1

Cable	Bolted PP	Resistive PP	Bolted PG	Resistive PG
C1	4.9%	3.6%	2.5%	4.1%
C2	5.1%	2.9%	3.9%	1.1%
C3	4.7%	6.1%	4%	2.7%
C4	1.1%	0.4%	2.1%	0.2%
C5	6.3%	0.6%	0.8%	0.6%
C6	5.3%	4.8%	6.2%	3.4%

In the Table 5-13 below, the performance of GP for fault location is compared against some other regression techniques, including Artificial Neural Network (ANN), Decision Tree, ε-SVM, and Nu-SVM. The comparisons are performed for Cable C2. As seen, GP renders higher accuracy compared to the other regression techniques.

Table 5-13: Comparison of different ML techniques for Cable C2

Fault Type	GP	ANN	Decision Tree	ε-SVM	Nu-SVM
Bolted PP	5.1%	8.6%	9%	8.2%	15%
Resistive PP	2.9%	4.8%	6%	5.9%	14%
Bolted PG	3.9%	8.9%	5.9%	7.7%	9%
Resistive PG	1.1%	2.9%	2.2%	2%	6%

5.2.6 AI Based Wide Area Control with Windfarm

In this section, an AI engine-based real-time wide-area damping controller design considering multiple synchronous generators and wind farms is discussed. The architecture is based on, a reinforcement learning (RL) Adaptive Critic Design (ACD) framework designed and tested to perform the optimal control of multiple generators based on energy function theory. The approach connects Lyapunov energy function theory with machine learning to design an architecture that damps inter-area oscillations.

5.2.6.1 Introduction

Real-time transient stability assessment and wide-area protection and control have been studied widely in the literature. Methods such as numerical integration and direct energy functions have been discussed and their advantages and disadvantages are mentioned in the earlier studies. Numerical methods are in general vulnerable to converge in real-time and direct energy functions, even though promising, are difficult to model without excessive simplifications. Measurement-based learning framework in this sense has a better advantage as such methods can be developed with AI and can be trained using offline and online techniques. This increases the chances of convergence as learning accurate but complex models through training provides accurate results with successive training approaches. RL-based offline and successive online training is a great candidate for such application without a prior knowledge of the learning objectives. The method can thus be used as a wide area system-centric controller and observer (WASCCO) and can be deployed based on a central learning framework (such as cloud computing) for offline learning and a grid edge device for subsequent online training/learning.

5.2.6.2 Problem Statement

Rotor angle stability theories focus on the machine angle oscillations and transient energy that creates such oscillations at the advent of an event such as a fault. Transient energy interactions and instabilities are caused due to the power mismatch between the generator's electrical and mechanical power. This can be captured in the form of a swing equation. During internal oscillatory events such as a fault, the machine rotor angles oscillate. Modeling of the generator dynamics can be presented classically using a third-order model of the conventional generator including the generator angle, speed, and excitation voltage equations with a Center of Inertia (COI) as reference. Then such energy functions can be related as a Lyapunov function such that a cost function that is proportional to the square

of the generator speed and voltage deviation can be defined. Minimizing the cost function then means minimizing the deviations in the voltage and speed in turn minimizing the transient energy that is generated during an abnormal event such as a fault. From the third-order dynamics of the generator including the excitation system, an energy function as a cost function can be developed where J represents the total cost function. The goal of the RL-based AI is to learn this cost function from the deviation in voltage (ΔV) and speed (Δω) measurements as x(t). The output u(t) is the voltage or power reference. The overall process is illustrated in Figure 5-42.

5.2.6.3 Machine Learning Process

The process of intelligent system construction starts with developing a database for training three neural network engines viz., Wide Area Neural network identifier (WANNID), critic neural network (Critic NN), and an action neural network (action NN). These three neural network engines conform to the family of ACD's.

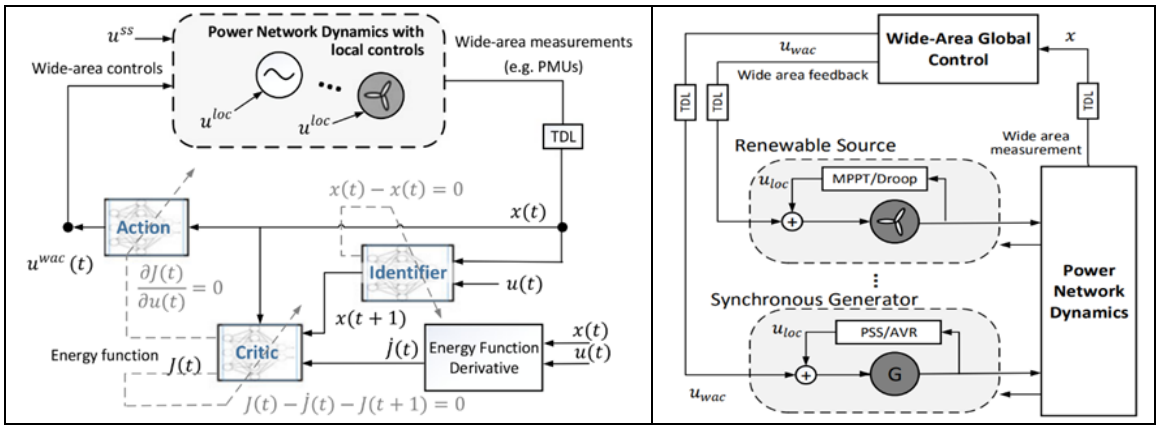


Figure 5-42: Wide Area System Centric Controller (WASSCO) design [100]

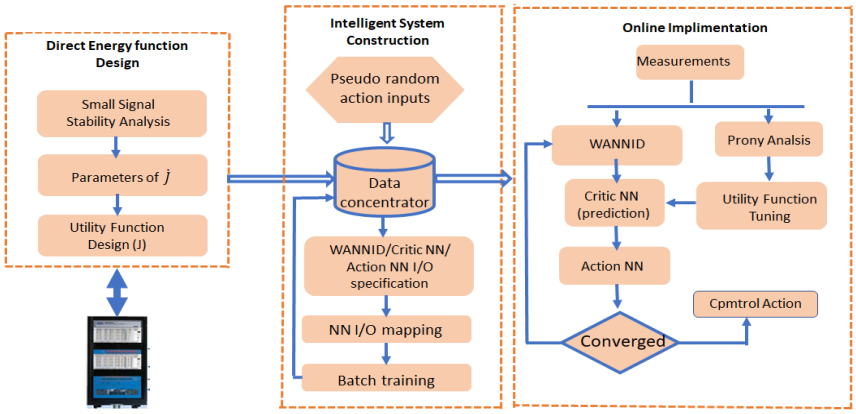


Figure 5-43: Reinforcement learning process [101]

The goal of ACD is to learn the Hamilton-Jacobi-Bellman equation based on optimal control theory through the Critic NN and find the right control or policy signal through the

action NN. Learning is done through optimization over time. ACDs are developed as parametric structures that can optimize over time including noise. During offline training, the goal is to minimize the estimated cost function J. For this purpose, three Feed Forward Neural Networks (FFNN) are designed. The output of each network is computed by calculating the inner product between the Weights (W) and the state-dependent vector (ϕ). WANNID develops the model of the system based on measurement, CriticNN optimizes the cost function and critic the policy, and Action NN develops the control signal. This process is repeated until convergence is achieved. Action NN is connected to the generator's exciter. In the online training phase, the training for WANNID, CriticNN, and ActionNN is performed sequentially in that order.

5.2.6.4 Implementation and Testing

The implementation platform is illustrated in Figure 5-46. The offline training is performed using a standard intel computer based on the model output from the real-time simulator. Once the offline training is completed, the controller is deployed on an edge device (TI controller). The measurements are compiled using the PMU and data concentrator. The online training process is based on the data streamed from the concentrator.

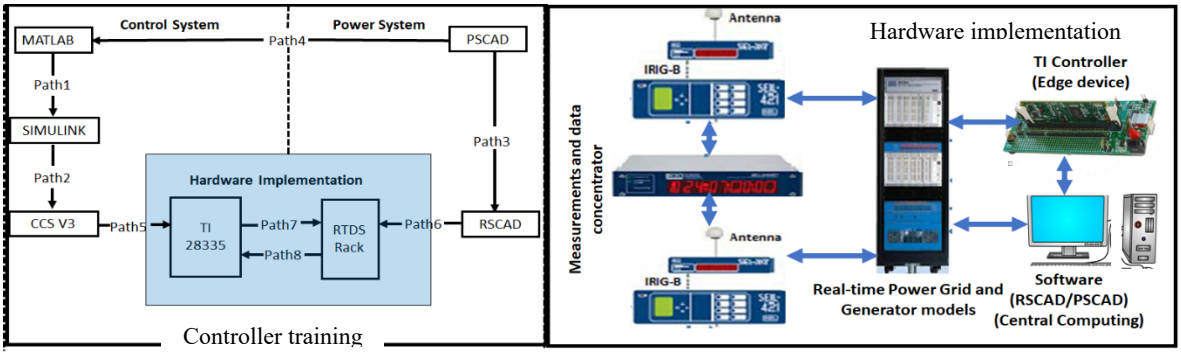
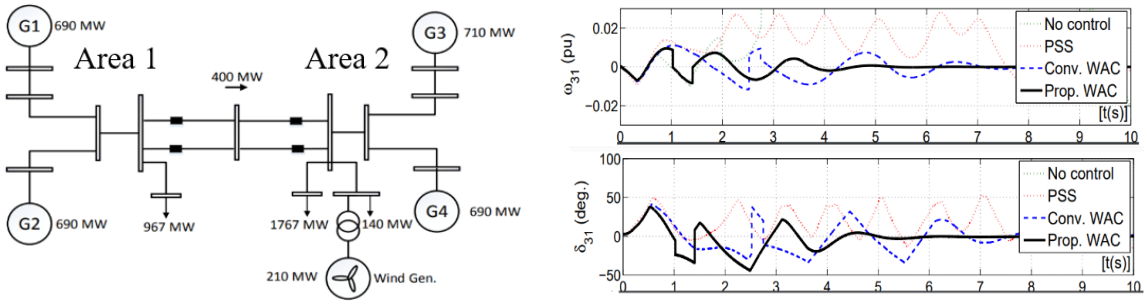


Figure 5-44: Implementation platform



a) Power grid model with wind farm b) Oscillations in angle and speed

Figure 5-45: Visualization for a two-area power grid with fault on area 2 [100]

For visualization purposes, a self-clearing fault in area 2 is illustrated in Figure 5-45(a). The fault is kept for 30 ms. The dynamics of the rotor angle and speed of one generator are illustrated in Figure 5-45(b). The illustration shows oscillations for four scenarios, a) with no damping controller (labeled as No control) b) with a Power System Stabilizer (labeled as PSS), c) an optimal control with the mathematical model of the system (labeled as Conv. WAC) and d) with the ML-based wide-area control (labeled as Prop. WAC). It can be seen that with suitable ML architecture and training the damping controller can outperform the conventional WAC and PSS.

5.2.7 Transient Instability Predictions

5.2.7.1 Introduction

Although power systems are designed to recover after being subjected to various disturbances such as faults, loss of generation or large loads, they are not immune to every disturbance which they may encounter. Large disturbance rotor angle instability events appear as aperiodic angular separations due to insufficient synchronization torque. Fast detection of impending rotor angle instabilities can allow taking automatic emergency control actions such as load or generator shedding to avoid potential system wide instabilities or blackouts.

5.2.7.2 Problem Statement

The typical local measurement-based out-of-step protection provided by distance relays has a limited capability of identifying an area wide disturbance. This shortcoming is typically addressed by using wide-area special protection system (SPSs). These systems are usually event-based systems designed to directly detect pre-identified outages which lead to rotor angle instability using status signals and initiate predetermined remedial actions. Implementation of an event based SPS can be complex, cumbersome, and costly due to the necessity of large number of teleprotection systems to convey all the system contingency information.

There is a potential for developing simpler response based wide area protection systems using synchrophasor measurements and Machine Learning techniques. Typically, the generator rotor variations are used to assess the rotor angle stability. The post-fault generator voltage magnitude and speed measurements also contain valuable information about the stability status after the disturbance. However, it is difficult to develop a simple physics-based relationship between the stability and the post-disturbance generator voltage magnitudes, rotor angles, and the speeds without resorting to full time domain simulations. Thus, it is useful to explore the potential of using system wide synchronized measurements and AI/ML techniques for predicting large disturbance rotor angle instabilities.

5.2.7.3 Rotor angle stability prediction scheme

Post-fault variations of generator voltage magnitudes, rotor angles, and frequencies of a power system subjected to a three-phase short circuit are in Figure 5-46(a). These waveforms are obtained by simulating a fault in the 39-bus New England test system. The

fault is applied at 0.1 s and cleared after five cycles, and the system exhibits rotor angle instability after the fault.

It is hypothesized that variation of voltages, phase angles and frequencies immediately after clearing the fault can indicate whether the system is going to be transient stable after the disturbance. Thus, it is possible to find a relationship between the observed variations and the transient stability status. For a multi-machine system, this relationship is not straightforward, but can be learned from examples using machine learning techniques. In this study, a binary classifier is used to learn this complex relationship. The arrangement of a transient stability status prediction scheme based on this principle is shown in Figure 5-46(b) and (c). The inputs to the classifier are the sampled values of the predictor variable, which could be the generator rotor angles, speeds, or the voltage magnitudes, while the output is the stability status. The algorithm is triggered by the voltage dip due to a fault, monitors the bus voltage recovery profiles following the clearance of the fault by the action of local protection relays, and collect the synchronously measured input features to the classifiers [102].

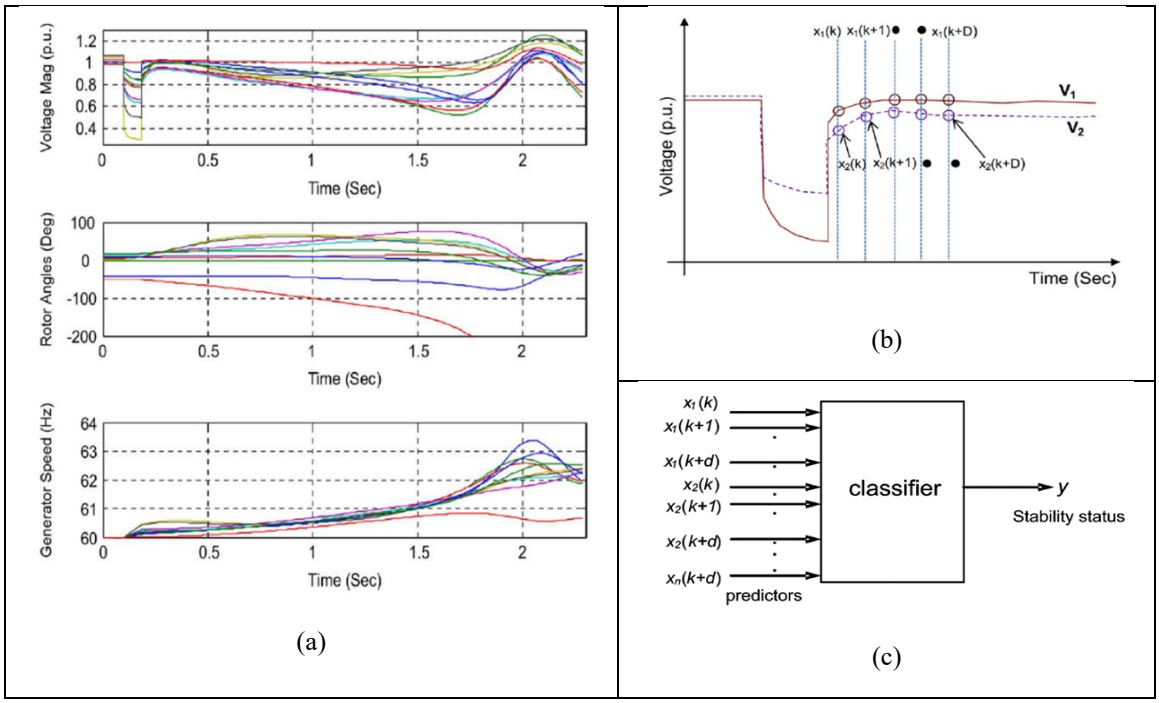
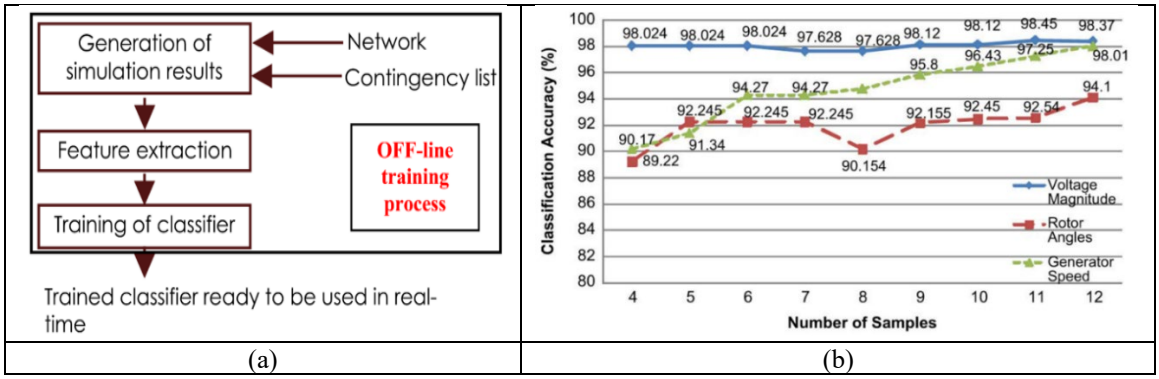


Figure 5-46: (a) Variations of the voltage magnitudes, rotor angles, and generator speeds during a contingency (b) Synchronized sampling of signals using PMUs, and (c) Arrangement of the transient stability prediction scheme

5.2.7.4 Application to IEEE 39 bus test system

A Support Vector Machine (SVM) classifier is trained to predict the transient stability status after a fault. The training and testing data for the IEEE 39-bus system is generated using dynamic simulations in PSS/E software, considering a large number of pre-fault operating points and contingencies. The process is shown in Figure 5-47(a). The results of

on-line prediction accuracy with testing data are shown in Figure 5-47(b). The graph shows the variation of accuracy for testing data with the number of sample points considered in the input data window.



(a) (b)

Figure 5-47: (a) Training process (b) Rotor angle prediction accuracy with different predictors and data window lengths.

The samples of all variables are measured at a rate of one sample per cycle (60 fps). Each point on each line corresponds to a different classifier configuration. For example, the SVM corresponding to the first point on the rotor angles curve takes 40 inputs (4 samples of 10 generator rotor angles) while the SVM corresponding to the last point on the rotor angles curve takes 120 inputs (12 samples from 10 generators). The best accuracy was obtained with the classifier which used the voltage magnitudes as inputs. This classifier achieved over 98% accuracy with just four consecutive samples of each generator bus voltage. The generator speeds are also good predictors of transient stability, but to achieve an accuracy comparable to that achieved with generator voltage magnitudes, 12 consecutive samples were required. The sooner the prediction is completed, the longer the time available to take control actions to avoid a possible system collapse, thus voltage magnitudes are the best predictors for this application.

The classifier that uses voltage magnitudes is further tested and some results are presented in Table 5-14, Table 5-15 and Table 5-16. For these tests, the system was simulated using PSCAD and contingencies included unbalanced faults. The classifiers were applied phase-wise, and a data window of just four samples were considered.

Table 5-14: Overall prediction accuracy

Condition	**Prediction Accuracy on Testing Data**	
	Classified as Stable (%)	*Classified as Unstable (%)*
Stable Case	100 % (660/660)	0 % (0/660)
Unstable Case	2.84 % (5/176)	97.16 % (171/176)

Table 5-15: Prediction accuracy under topologies not contained in training data

Scenarios	Prediction Accuracy on Testing Data		
		Classified as Stable (%)	*Classified as Unstable (%)*
-Generator 37 out -Line between Bus 25 and Bus 26 out	**Stable Case**	99.76% (430/431)	0.23 % (1/431)
	Unstable Case	3.15% (4/127)	96.85% (123/127)
-Line between Bus 5 and Bus 8 out	**Stable Case**	100% (446/446)	0 % (0/446)
	Unstable Case	3.57% (4/112)	96.42% (108/112)
-Line between Bus 22 and Bus 23 out	**Stable Case**	100% (454/454)	0 % (0/454)
	Unstable Case	3.84% (4/104)	96.15% (100/104)

Table 5-16: Prediction accuracy with noisy data after re-training the classifiers with noisy data

Condition	Prediction Accuracy on Testing Data	
	Classified as Stable (%)	*Classified as Unstable (%)*
Stable Cases	98.41 % (1189/1202)	1.58 % (13/1202)
Unstable Cases	4.12% (43/636)	93.24 % (593/636)

5.2.7.5 Conclusions

Transient stability can be predicted accurately and fast using post-fault bus voltage magnitudes. According to Table 5-14, the scheme has 100% security, and shows 97% accuracy in correctly predicting unstable cases. The noisy data reduces the accuracy, but when the classifiers are re-trained with noisy data, accuracy improves. The classifiers are able to predict the instabilities with over 95% accuracy, even under topologies that were not included in the training data patterns. The study also showed that prediction accuracy increases when the loads include induction motors. Simulation studies with Venezuelan power system showed 100% accuracy [102]. A more sophisticated version of the classification scheme is presented in [103].

6 PRACTICAL APPLICATION CONSIDERATIONS

Once the acceptance criteria discussed in Section 7.3 is satisfied, there are several logical steps to be taken to implement protection and control using artificial intelligence. This is to allow that the grid incrementally is introduced to protection that is data driven and security, reliability, and resiliency are not compromised.

The applications can be in individual devices and functions, substation functions or system-wide functions. This part of the report will cover driver, benefit, industry use cases, algorithms, AI approach and potential challenges for various applications.

6.1 Application Implementation

Application implementation includes understanding the data types, sources, formats and management, defining the data structure, computational platforms, and learning framework, testing, and validation. Each of these are discussed in the following subsections.

6.1.1 Data Types, Sources, Formats, and Management

This section focuses on the sources of data for AI and ML applications in protective relaying as well as data formats used by the data sources and common methods used to manage the data. In the text below, the word ‘sensor’ is used as a generic term that encompasses the IEDs, merging units, PMUs, etc.

6.1.1.1 Data Types

Sensors that are deployed in an electric power system can record many different data types, such as waveforms, rms samples, phasors, and more.

6.1.1.1.1 Waveform Samples

Waveform measurements record short periods of three-phase instantaneous voltage and/or current and can have varying sample rates (Figure 6-1). They can be triggered by programmed algorithms detecting electrical disturbances or scheduled periodically. Most measurements have defined start and finish times based on the triggering algorithm. Some sensors offer continuous waveform recording and may use lossy compression and a FIFO buffer. Waveform sampling can be synchronized to either an external clock or the current system frequency. Additional channels, such as neutral current, and derived values like residual current and system frequency may be included. Alongside waveform samples, floating point values of intermediate or final values of relay targets or flags, derived from the samples, may also be provided at the same sampling rate.

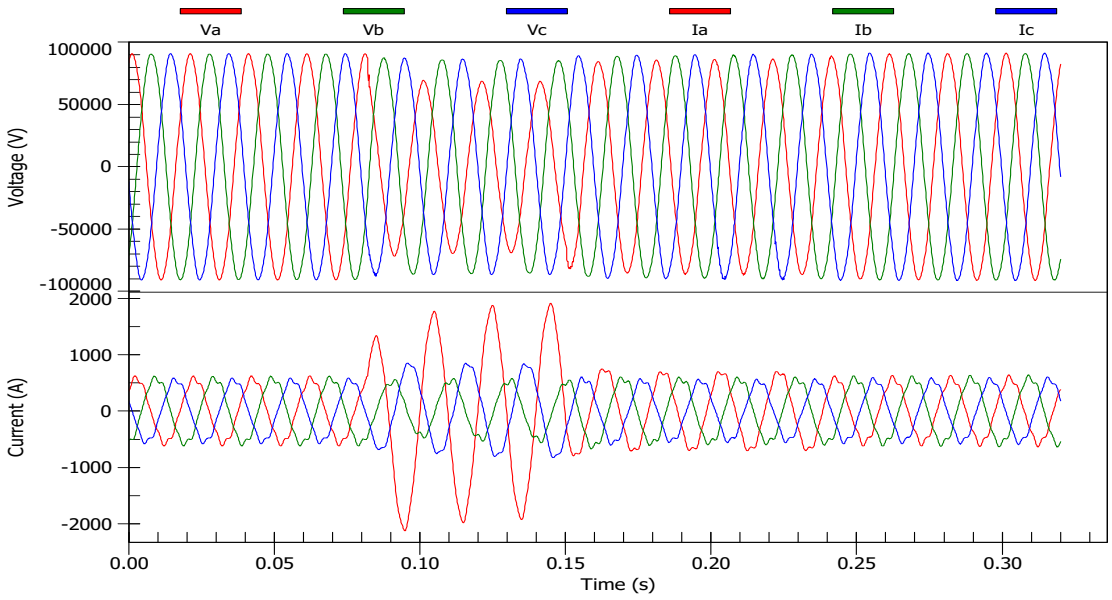


Figure 6-1: Example Waveform Chart Showing Voltage Samples and Current Samples

6.1.1.1.2 Digital Status Values

Many sensors that record waveforms or rms samples also record digital status values. The values can be used to represent a time series that has two states, such as on/off, high/low, true/false, or 1/0.

Figure 6-2 provides an example with voltage and current waveforms and three digital status values (Feeder 05, Feeder 06, and Feeder 10), which provide the state of the breaker for each of these three feeders. The digital status for Feeder 05 shows a thin line when the associated circuit breaker is closed and a thick line when the associated circuit breaker if open.

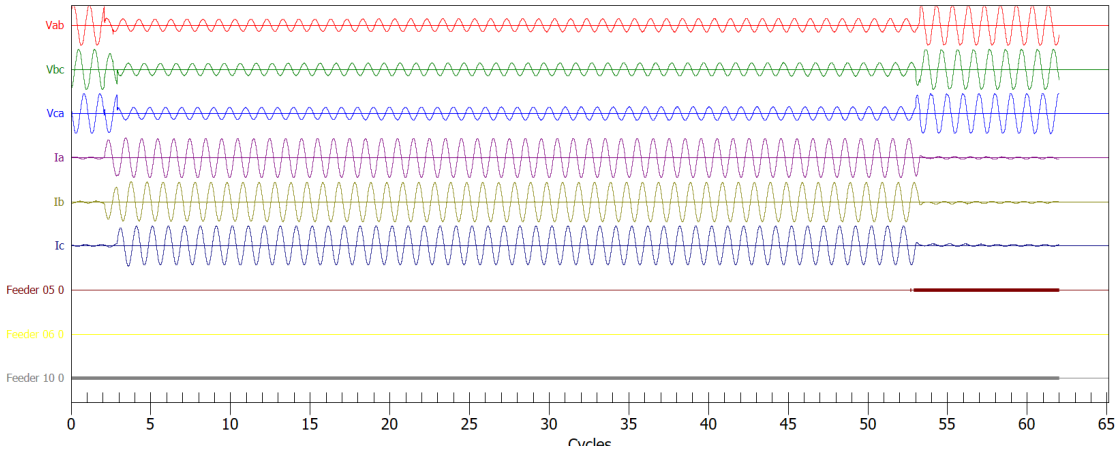


Figure 6-2: Example Chart Showing Digital Status Values with Voltage and Current Waveform Samples

6.1.1.1.3 RMS Values

RMS values can be recorded due to a fault or disturbance at the same time as waveforms or in place of waveforms. They can be comprised of the root mean square (RMS) value of a single cycle or can be the averaged over a window such as 2 cycles, 3 cycles or 200 ms (as per IEC 61000-4-30). Figure 6-3 is an example chart showing the voltage and current RMS values derived from the waveform in Figure 6-1.

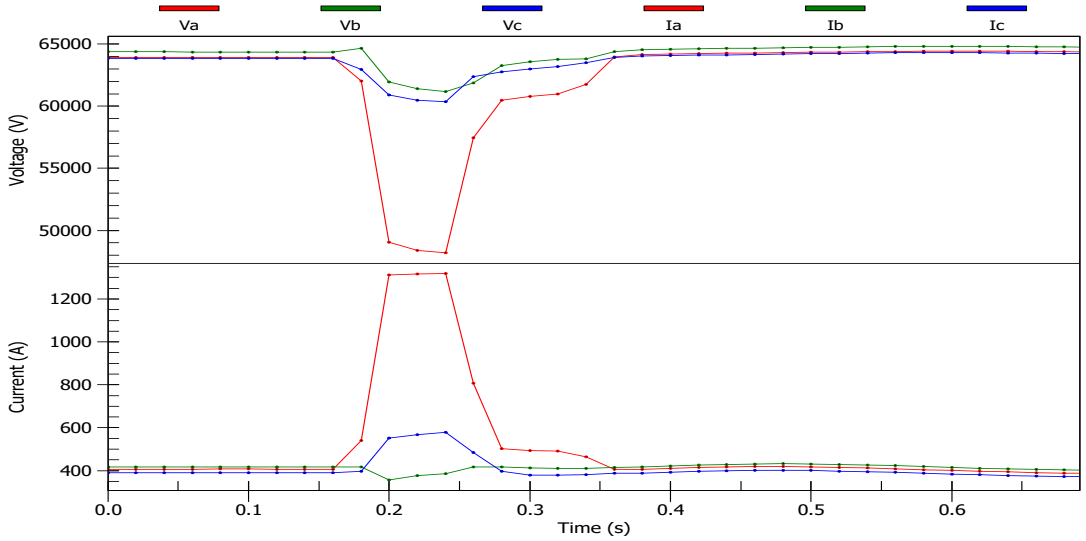


Figure 6-3: Example RMS Values Chart with Voltage and Current Samples Derived from Waveform from Figure 6-2

6.1.1.1.4 Phasor Values

Phasor values can be recorded to summarize the magnitude and/or phase angle of voltage and current values. Phasor values are usually derived for the fundamental frequency component of the power system (e.g., 50 Hz or 60 Hz) but they can be derived for harmonics of the power system (e.g., 3rd harmonic for a 50 Hz system or 150 Hz). Figure 6-4 is an example showing three-phase voltage and current phasor values.

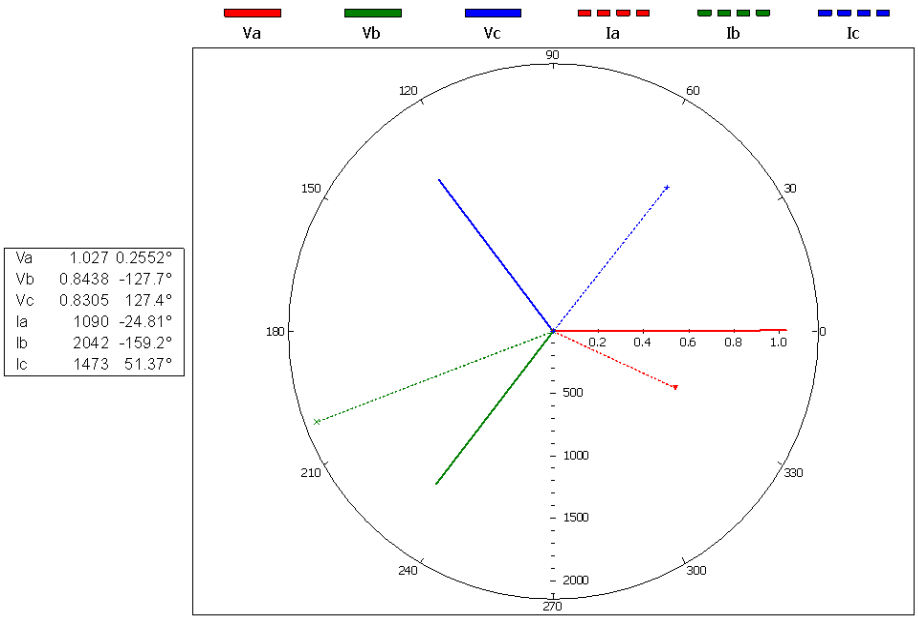


Figure 6-4: Example Phasor Chart Showing Voltage Phasor (Per Unit) and Current Phasors (Amps)

6.1.1.1.5 Time Series (Data Logs)

Time series records a range of basic and advanced electrical quantities, such as rms voltage, rms current, phasor magnitude, phase angle, line frequency, power factor, real power, reactive power, apparent power, total harmonic distortion, and harmonic magnitudes. Sensors can be programmed to provide these values at regular intervals, like every 15 minutes, 10 minutes, or even every second. They may also record values at a faster sampling rate during specific events, like when rms voltage drops below a threshold.

To save storage space, some sensors or their associated data collection gateways or servers use lossless or lossy compression algorithms. Lossy compression may include settings that determine the allowable loss of precision in historical measurements.

Sensors can store either instantaneous values in a time series, such as the rms voltage for one cycle at a specific time, or statistical summaries at regular intervals, like the minimum, average, and maximum voltage every ten minutes. Values are typically recorded and stored by phase (e.g., Phase AN, Phase BN, and Phase CN), but some meters might average line-neutral voltage/currents or line-line voltages to save on memory or file space. Time series can display values for different voltage phases, as exemplified by Figure 6-4 which shows

RMS voltage for Phase AN, Phase BN, and Phase CN.

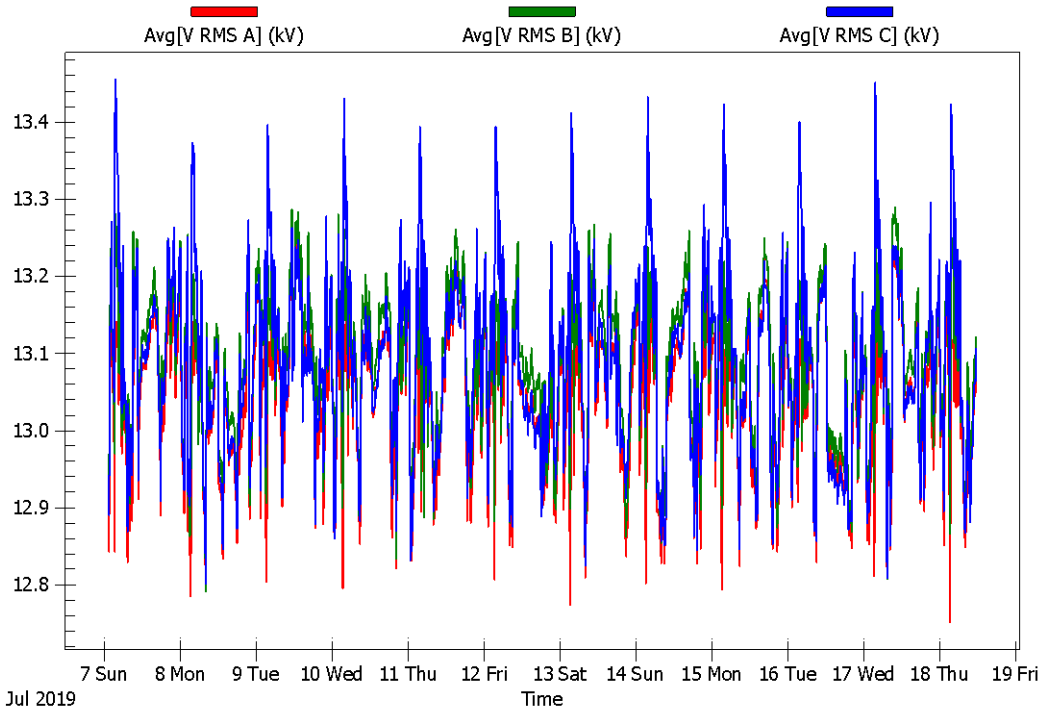


Figure 6-5: Example time series chart for RMS Voltage on Phase AN, Phase BN, and Phase CN

6.1.1.2 Event Categorization

Waveforms, RMS values, and phasor charts can be categorized using many IEEE standards or custom codes.

For example, IEEE Std 1782 [109] provides categories and subcategories for the cause of an interruption (e.g., vegetation, lightning, wildlife, weather, equipment, etc.) IEEE Std 1159 [110] provides broad categories related to power quality (e.g., momentary interruption or instantaneous voltage sags). IEEE Std C37.114 [111] provides descriptions of fault types as single-phase-to-ground, phase-to-phase, phase-to-phase-to-ground, and three-phase.

A subject matter expert may review the waveforms, RMS values, and phasor charts and categorize them due to their cause (e.g., cable failure or load tap changer failure) or their source (transmission system distribution system, customer system, neighboring electric system, etc.)

Cause information may also be extracted automatically from an electric utilities outage management system (OMS) or distribution management system (DMS).

6.1.1.3 Data Sources

Waveforms, rms values, or phasor values can be recorded by a number of different sensors:

- Microprocessor Relays
- Digital Fault Recorders
- Remote Terminal Units (RTUs)
- Power Meters
- Energy Meters
- Power Quality Monitors
- Continuous Waveform Recorders
- Sequence of Event Recorders
- Lightning Detection Systems
- Manually Supplied Information to a Utility Automation System (e.g., the cause of an interruption as reported by a customer)
- Weather sensors

Not all aforementioned sensors record all data types.

6.1.1.4 Data Formats

Data from these sensors can be stored in file formats

- IEEE Std C37.111 COMTRADE Files: Waveform Samples, RMS Samples, Phasor Values, Digital Status Values, Time Series
- IEEE C37.118.1 Synchrophasor: Phasor Magnitude and Phase Angle, RMS Values
- IEEE Std 1159.3 PQDIF Files: Waveform Samples, RMS Samples, Phasor Values, Digital Status Values, Time Series
- Text Files: Custom CSV Files, Custom Sequence of Event Logs

6.1.1.5 Data Management and Storage

Waveform samples, digital status values, rms value, and other values can be stored in a folders and subfolders. For example, a simple system may employ folders and subfolders of IEEE COMTRADE Files or IEEE PQDIF Files.

Some data users employ commercially available or proprietary databases or data warehouses to store the measurements in a relational database or time series historian.

When storing data in a cloud-based platform, additional storage options include data lakes.

6.1.2 Data Structures

6.1.2.1 Introduction

Data structuring and organizing is critical in any AI/ML based application. The fundamental concept of data structures is about how data is managed to perform tasks like organizing, storing, accessing, and editing. In modeling problems, the management of data

includes the relations or connections between data most often developed using complex structured database.

Basic data structures are single variables, vectors and arrays. The focus of this kind of structures is to access the data using position indices, normally the numbers of rows or row and columns in an ordered sequence.

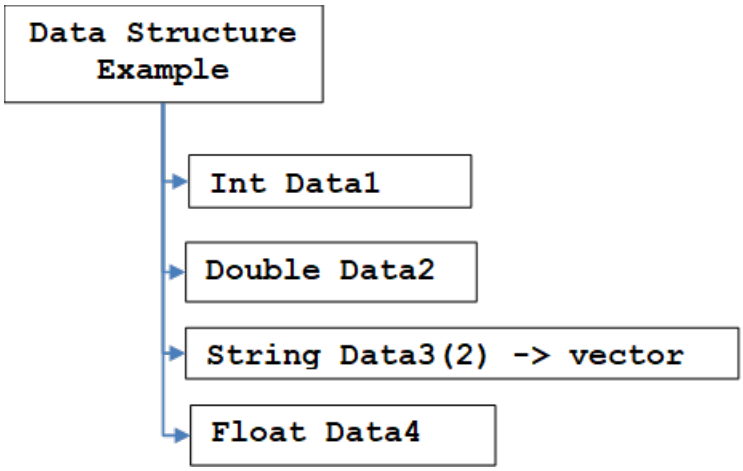


Figure 6-6: Example of a data structure

With the development of Object-Oriented Programming and modern programming languages, data types were expanded to consider a combination of several types of data. For example, it enables the capability of storing a string and numeric value in one variable structure like an array that uses a key to access data or even to include properties of the data (dictionaries). Other concepts like sets or containers were introduced by the evolution of programming and with this expanding the concept of programming using close to real world structures. Currently depending on the programming language used, there are many different data structures that can be considered when developing an artificial intelligence application. In this section key factors that needs to be considered while using data structures for application of AI/ML are discussed.

6.1.2.2 Defining Data Structures

According to the algorithm or techniques to be implemented, data structure can be defined based on the attributes and specific data to be used. For example, an algorithm that uses population principles needs data about the size of population and solution fitness (objective). An approach that can be used to define a data structure is to perform a manual derivation of the algorithm such as checking the required variables, and how the data can be packed in structures for running the algorithm in a high-level way. Relating this to AI application, assume that a genetic algorithm for an optimization application of an AI problem needs to be implemented. It is necessary to define the structure for storing the solution considering the use of an identifier (id) for every solution and include among the data attributes the value of the fitness function (objective).

Special attention should be paid to the performance requirements when data structures are used. By principle, the more complex are the data structures, the worse the performance becomes. Therefore, the size of data structures inversely proportional to the performance of the algorithm. The performance can be optimized considering the design principle of minimum computation cost by processing only data structure parts that are fundamental for algorithms execution. It is highly recommended to test the performance of data structures by the execution of a draft algorithm implementation before totally defining the data structure to be used in an algorithm.

6.1.2.3 Data Management and Relationships

The first aspect to be considered is the level of generalization of the problem to be solved and input capabilities. When the level of generalization (capacity to process several kinds of inputs – problems) is higher and data structures are bigger and complex, it is very important to analyze limits of data to avoid algorithm collapsing due to large quantity of data. Data segmentation and control the size of data structure are needed based on the application in hand to avoid crashing during the execution. Second, determining correct volume of data is critical. For this studying the capability of programming language to define the data structure is needed, followed by an evaluation of the size of the data in the structure and its impact on the algorithm's performance. Special attention is required when dynamic data structures are used. Control of size and segmentation may be needed to have acceptable algorithm performance. Third, the data relationships will be studied well based on the application when complex algorithms are designed. Understanding the data relationship provides access to the right information, helps avoid passing all data to a process with significant loss of execution time, and it allows for more efficient programming to scale the models in the future. A key point that needs to be considered is the usage of identifiers in a simple and consistent way to avoid data duplication and confusion. A best practice relating to data structure relation definition is to include data attributes in data structures to reduce the computational burden. It is also important to think about how the program or algorithms will be maintained, how easy will be to scale and how feasible is the algorithm to obtain the desired performance and user friendliness.

6.1.3 Computational Platforms

Computational platforms in power system protection and control are formed by hardware, software, and communication systems for data collection, processing, and dissemination. The selection of a specific computation platform is highly dependent on the type of AI/ML applications to be implemented, and how/where an AI/ML application is going to be implemented. These may be realized in various ways. One practical example of such a computational platform would be Centralized Protection and Control (CPC) systems within a substation [104]. CPC systems are modular and collect data from all the devices within a substation. Due to the modularity of the hardware, CPC platforms can be expanded as needed to provide computing power for AI/ML-based applications. CPC systems are great computation platforms for AI/ML-based applications as they have access to substation-wide real-time data for learning and inference. If employing expanded communications schemes, CPC systems may also obtain additional data from neighboring substations, edge devices in feeders, or from the cloud, giving rise to a hybrid edge/centralized system.

Advanced high-impedance fault detection and location is an example of a challenging issue that could be addressed with an AI/ML-based algorithm running on a CPC system. High-impedance fault detection algorithms are difficult to implement in traditional asset-specific IEDs due to the computational power requirements and the extensive requirement of on-line training. If on-line training is moved to the cloud, with data sharing from and to field devices (such as reclosers, fault indicators, or feeder V/I sensor monitors), then less computationally intense algorithms can run on the edge. Still, though fault *detection* is possible on IEDs, fault *location* remains difficult with the limited environmental information available at the edge. Security and dependability can both greatly be improved in a centralized system. Using a CPC, information and fault detection from all the algorithms on the edge can be combined. In addition to improved data, more computationally intense algorithms can be employed in the centralized system to verify the results received from the edge. The location and relative separation of the edge devices help give the CPC a clearer picture with which to resolve the problem of fault detection *and* location.

6.1.3.1 Hardware

6.1.3.1.1 Hardware Options

To effectively run the machine learning models once they are built in a production environment, where the models will reside is important to consider. This decision will be different for different utilities, depending on the existing architecture of the backend data repositories/software, cybersecurity practices, communication infrastructure, and general preferences. There are three different options for storing the operational machine learning models: hardware on the edge, centralized servers, and the cloud.

6.1.3.1.1.1 Edge Hardware

There are many solutions available for edge processing. However, edge processing sometimes could be limited in computational power depending on the device chosen. Some vendors offer all in one solution where they provide sensing in addition to computational power. Such hardware may reside in the field or in a substation.

6.1.3.1.1.2 Centralized Processing

Centralized processing is more powerful computer located inside a facility. It enables to collect data from various sensors across the system and build/run machine models on a larger scale.

6.1.3.1.1.3 Cloud Computing

Cloud computing is an efficient approach to collect data from all sensors, process it, and use it to build ML models without the need for hardware. There are many vendors that offer cloud computing/data storage services safely.

6.1.3.1.2 Hardware Considerations

There are many considerations that will help identify the hardware needed and the architecture (Edge vs Central vs Hybrid) to choose. Some may include:

- Location of the sensors/devices collecting the data relative to each other. If they are all located in the same area, one might choose edge computing

- Amount of data collected from sensor. The larger the data, the more consideration one would make for some level of edge processing
- Number of data sources that model relies on. The more data sources contribute to the model, the greater the need for either a centralized computing resource or cloud computing.
- Cybersecurity

6.1.3.2 Software

6.1.3.2.1 Software Options

The software space for machine learning can be very confusing. There are new companies being created every year that claim to provide the best software to develop and run machine learning models. It is therefore imperative that utilities be very clear on their objectives for what a software platform should be provided. Their understanding of the end goal will enable them to identify the level of UX/UI interface that is needed depending on who the end user is.

6.1.3.2.2 Software Considerations

There are many considerations that will help identify the software needed to deploy the models and view the results

- Software needs to comply with cybersecurity/IT standards and requirements
- Software needs to provide adequate level of visibility and flexibility that is appropriate for the end user
- Software is scalable to accommodate the exponential amount of data without sacrificing speed, reliability, etc.
- Software adapts to the machine learning models language, libraries
- Software is able to provide strong visualization that is specific to utility applications and not too generic

6.1.4 Testing and Validation

6.1.4.1 Learning. Validation and Testing in Machine Learning (ML) Architecture

There are two different ways the ML architecture uses the data, a) offline or batch machine learning b) online machine learning.

In the batch machine learning process, the data is first gathered and divided into training and validation sets. Then the feature set has been extracted and further the features from the set have been used for training including the data labeling. This is used as the input to ML/ANN. The outcome is the learning model that has been developed based on the training. This model is then used for validation and testing. For validation purposes, part of the data from training is used which is then evaluated using a different set of data called testing.

Once the training and testing of the batch learning are complete then the model can be used in production (for the problem that needs to be solved such as prediction).

In the online training, ML algorithm/ANN is connected to the streamed data from a system through a feature set, feature extraction and data labeling. With the online training, the learning model continue to learn and update with the help of the ML algorithm so that the new scenarios that have not been trained can be learned in this process.

Similarly, the prediction can be performed with the previously acquired data namely offline prediction or using the streamed data namely online prediction.

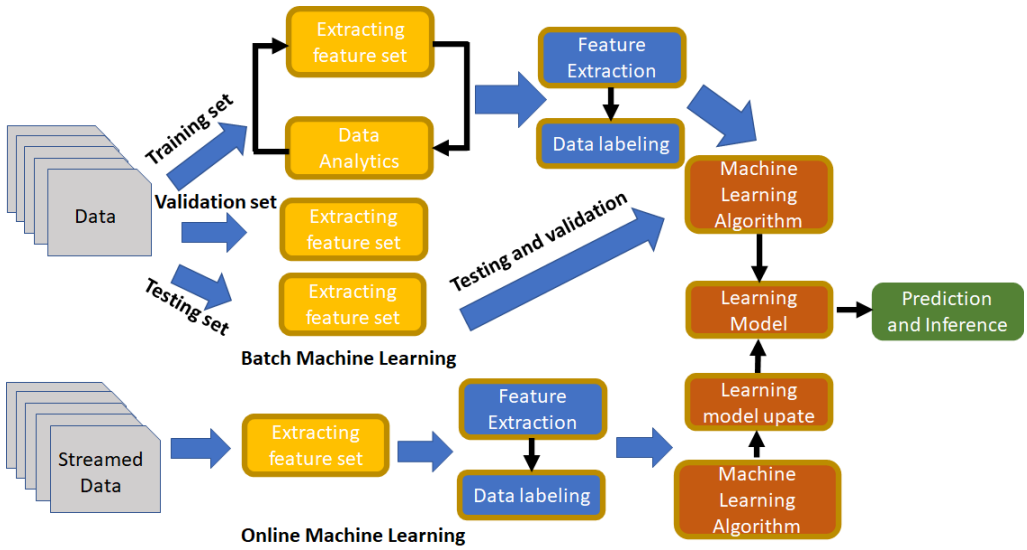


Figure 6-7: Coordinated learning framework based on offline/online machine learning

Figure 6-7 illustrates the process of combined offline and online training. The offline training develops a learning model that can predict or infer the possible outcomes. The model can be updated using an update algorithm evolved from the offline trained machine learning algorithm. In the process of independent online learning without offline counterpart, the learning starts from an initial condition and learning model updates are performed from the initial state.

6.1.4.2 Testing

Testing in the context of an AI/ML system pertains to the process of ascertaining the performance of such system. There are two possible aspects to this:

1. Ensuring that the AI/ML system exhibits satisfactory with respect to the error rate requirements. The required error rates can be imposed on each type of errors individually (e.g., false positives, false negatives), or combined.
2. Ensuring that the computational environment in which the AI/ML system is deployed has proper resources to accomplish sufficient data acquisition in the field, on-line inference, or on-line training.

In this section the process of offline learning is discussed in detail. Preparing the system to perform classification or estimation in the field is referred to as **training**, while the classification or estimation in the field is referred to as **inference**.

6.1.4.3 Parameter and Hyperparameters

Without any loss of generality, the following discussion assumes an AI/ML system that needs to perform a task of classification. In this context it is further assumed that the training of the system is performed in a supervised manner: data that describes observations in the field is available for training along with the correct labels for each observation. In general, every AI/ML algorithm has a set of direct parameters that are set through training, using acquired data. Once the architecture for the algorithm is selected, these direct parameters are produced through optimization, forcing the algorithm to try and conform to the labels for each of the data points.

Take an ANN as an example. Each such network can have several layers in the network, and several neurons per layer. Neurons between layers can be fully connected, or scarcely connected. Each neuron can have various activation functions. There are other aspects to take into considerations as well. The totality of all such aspects necessary to create an AI/ML algorithm is covered within the term architecture. It is easy to show that multitude of aspects can be defined for other ML algorithms as well. All the aspects within the architecture are referred to as **hyperparameters**.

It should be clear at this point that the training mentioned above, through optimization, results in optimal parameters for the model (given the architecture), while selection of hyperparameters is not obtained in the same manner. This means that once the architecture is selected, the available data can be used along with the optimization method to compute the optimal parameters, which in turn sets the performance of the system with respect to the error rates etc. as mentioned earlier. Thus, the only way to improve performance further, if the performance is not satisfactory, is to change the architecture, or hyperparameters.

6.1.4.4 Estimating Performance

Before looking into how to change the architecture, it is imperative to look into how the performance in the field can be estimated. Every AI/ML system is designed and trained using finite amount of available data. However, once the system is deployed, it is most likely to encounter data samples that had not been used in training. Therefore, it is imperative to assess the inference performance on the samples not previously used.

A common way to approach this problem is to split the available data into training and testing. In this manner the system is trained on a subset of data (training set), and its performance is tested on the complement subset (testing set), which accomplishes testing on the samples not previously seen.

Assume there are N samples available in total. Of these N samples, M samples are used for training, and N-M samples are used for testing. This selection is referred to as a **test/train split**. At this point we can use the M samples to train the model (optimize the parameter of the system) and assess the performance on the N-M testing samples. We can then randomly

choose a different subset of M testing samples, retrain the network on the N-M samples, and reassess the inference performance. This selection of splits can be performed multiple (K) times, and the performance can be averaged over all (K) splits. Once the assessment is performed, the system can be trained with all N samples before deployment, and the estimate of the performance should hold in the field. Figure 6-8 illustrates this process of selecting train/test splits.

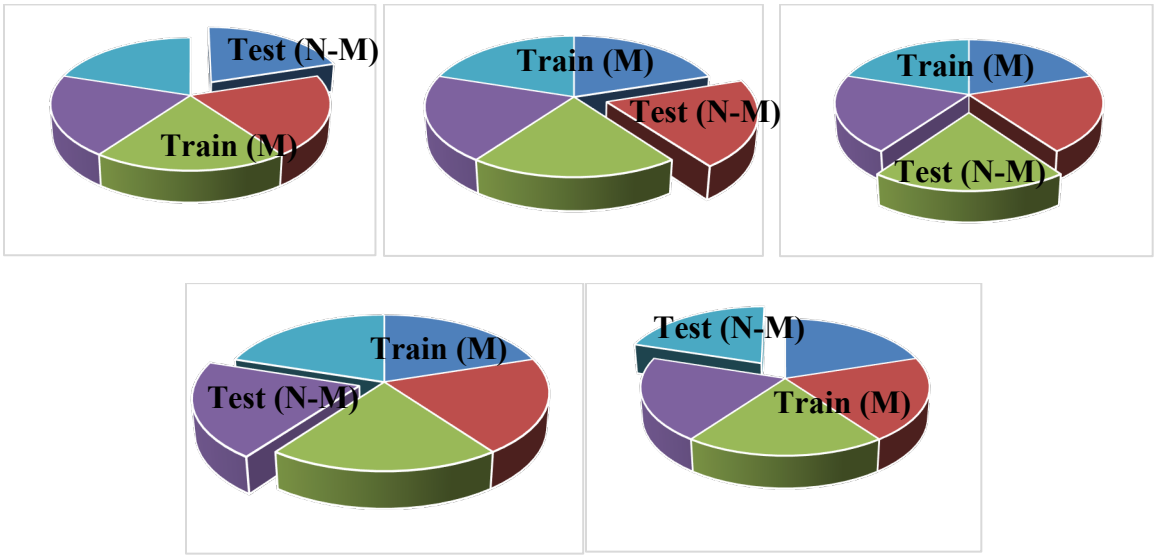


Figure 6-8: Train/test splits

Now consider a situation where the system was deployed and was forced to infer K samples from the field. As mentioned before, the samples encountered in the field are most likely samples that had not been used in training. However, the system was trained on N samples and asked to infer on K samples, as opposed to the earlier situation where the training was done on N-K samples and inference was done on K samples. It should be clear that the performance in the field would be better than the estimate in the paragraph above, because the system was trained on more data before deployment.

Note that in the training process, the testing samples were never used to change the architecture of the model, but only to assess the inference performance on "previously unseen" samples by the ML algorithm.

6.1.4.5 Cross-validation Loop

The development described thus far assumed that the architecture of the algorithm is not changed throughout the training process. However, changing the hyperparameters could improve the performance. Staying with the ANN example, maybe performance of the algorithm could be realized if the number of layers is increased, number of neurons per layer is changed, or activation functions per layer are changed. Searching for the optimal hyperparameters is not trivial and cannot be accomplished by using the same approach as optimizing the parameters of the algorithm. The process for optimizing hyperparameters is

most frequently performed by a grid search over possible combinations of hyperparameters and is illustrated in the Figure 6-9.

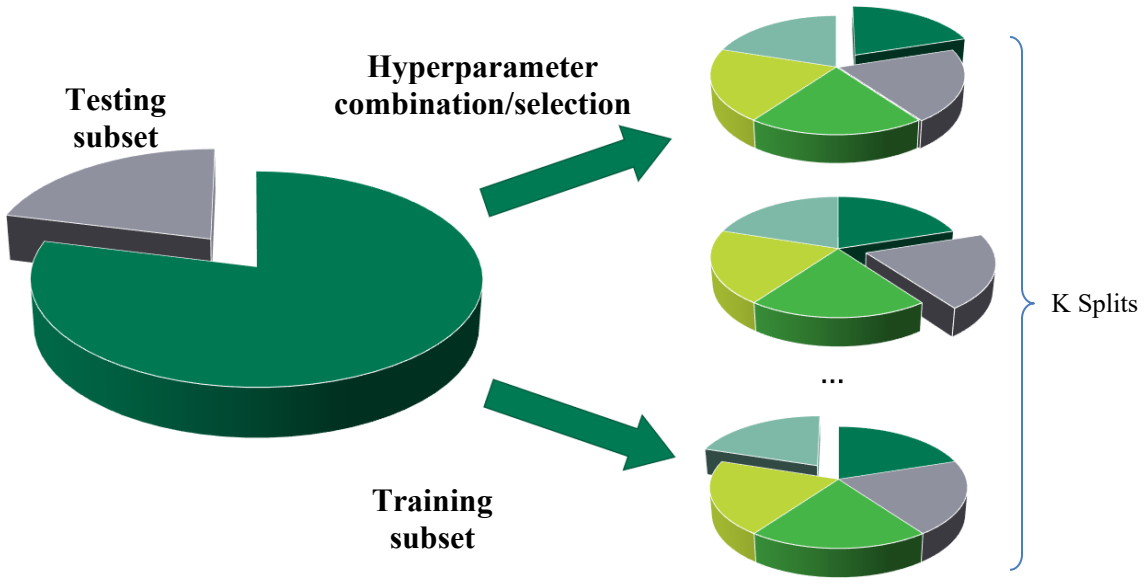


Figure 6-9: Cross-validation process

As Figure 6-9 shows, for each combination of hyperparameters, K splits are generated, much like in the process outlined in Figure 6-8, except the splits are now generated from the training subset (not the entire dataset), and the testing subset is left untouched during this entire process. The K splits are now referred to as **test/validation or cross-validation splits**, and the process is called **cross-validation**. For each combination of hyperparameters, performance is assessed by averaging the performance for each of the K splits. Optimal set of hyperparameters is the one that gives the optimal performance over the K cross-validation splits.

Once the hyperparameters are optimized, the final performance is assessed by training the model using the optimal architecture (hyperparameters) and the entire training set, and testing using the previously left-out testing set.

It is important to note that the testing set was not used in training at all, and it never should be. If the performance during the cross-validation is not satisfactory, the AI/ML model needs to be modified until the performance is adequate, without testing on the testing set. The reason is that once testing set is used to assess performance, that will be the end of the training process. If changes to the model are made as a result of observing the performance using the testing set, then the testing set is technically being used in the training, and that violates the paradigm of keeping the testing set separate (not previously seen by the mode). In this case the estimate of the performance becomes overly optimistic, as opposed to pessimistic.

Some AI/ML designers may still end up taking this (erroneous) step but mingling testing data in training is highly discouraged.

6.1.4.6 Field Implementation and Testing

Field implementation computing platforms can have multiple layers. The stored data from pre-acquired dataset is generally located in the cloud which enables offline training/learning. The cloud computing infrastructure necessitates all the different layers of offline training as discussed in Section 6.1.3. The prediction or inference engine can reside in the central location making the central computing cluster a decision-making entity. The inference or prediction is then delivered to field devices via a communication backbone (wired or wireless depending on the application and severity). Online learning can be performed on the edge server and decision control at the edge level can be augmented based on the streamed data and online learning.

Field deployment is generally performed after the offline training and thus online training is further facilitated based on real-world data streaming. With online learning, the architecture continues to learn such that the new information is used to extend the learning Euclidean space. For offline learning, the pre-acquired dataset is used. The overall framework is illustrated in Figure 6-10.

As with any protection and control scheme field implementations, the implemented AI/ML computing infrastructure also needs to undergo similar field-testing processes before putting into operations. This may include a period of running the scheme in a monitoring mode, and performing field testing using standard test equipment and procedure, etc.

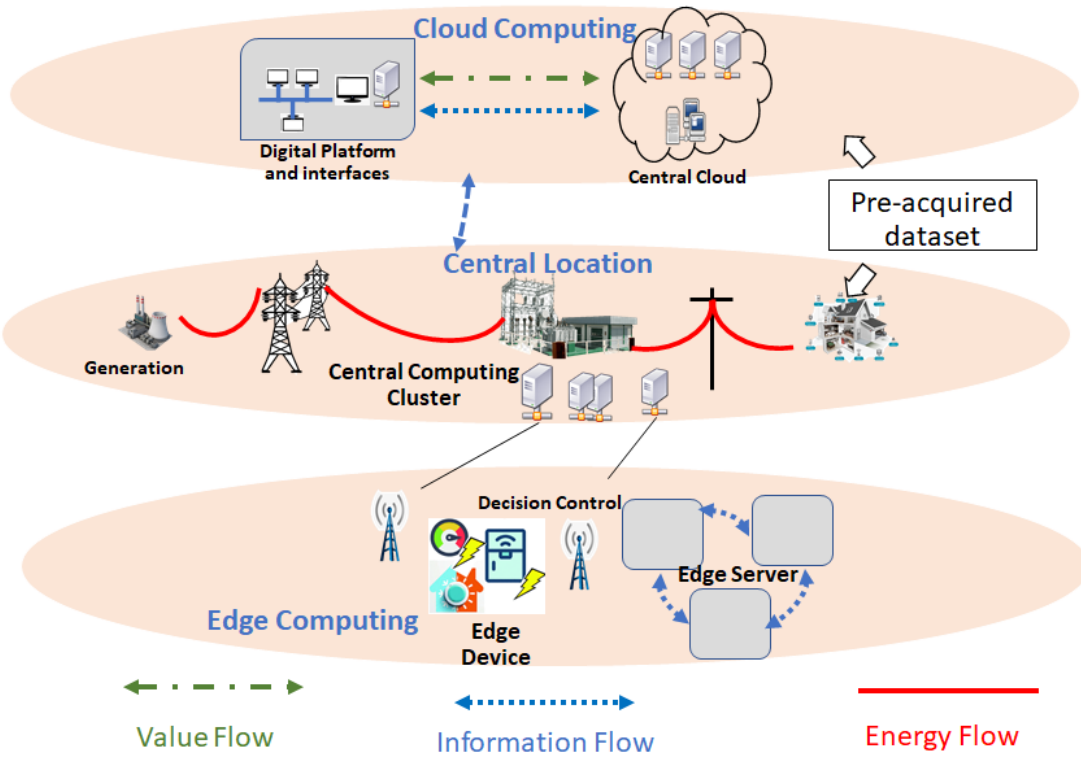


Figure 6-10: Generic Field Implementation Computing Infrastructure

6.1.5 User Training

As with any power system protection and control products, end users need to be adequately trained to use the AI/ML based protection and control products properly.

The following is a short list of topics related to training. Training modules could be self-paced or instructor led, hands-on interactive with the user.

Manufacturers: Prepare video-based (cloud-based) online training and/or in-person training modules from basic to very advanced levels would be prepared for both end users and supporting personnel. Training modules may include: AI/ML basics, installation/configuration/verification of the product, problem analysis / self-diagnostics and precision / pinpoint problem reporting, troubleshooting guidance, life cycle maintenance hands-on training, upgrade procedures and steps, acceptance testing, etc. Also offer both users group and individual self-paced training opportunities.

End-users: Identify line of businesses (LOB) to be trained (e.g., P&C, Engineering, Maintenance, IT, Security, Operation) is the first task. Trainees are to meet certain skillset requirements of the training modules. Types of training modules need to be selected based on LOB of the users to be trained which always include common modules such as security, life cycle maintenance process, etc., as well as how AI/ML P&C applications can supply reports needed for regulatory compliance (operational, maintenance, security, performance measures). Users could also learn using P&C database to track for firmware / multiple firmware versions for the P&C AI/ML based application products, and to determine metrics and methodologies to compare AI/ML applications with existing / conventional product or system performance.

7 Use of AI/ML for Protection: Risks, Challenges and Acceptance Criteria

7.1 Risks

Although there is clear evidence that the AI/ML technology could be successfully applied to solve practical power system protection and control problems, the AI/ML technology should never be considered as some type of “magic” technology that can be applied to solve all types of protection and control problems. It will be applied with the same type of care as many other technologies that have been applied for power system protection and controls. Otherwise, the application of the technology could inadvertently introduce new risks to reliable power system protection and controls by:

- Applying the technology to solve the problem that is not suitable for it
- Failing to recognize the major challenges in applying the technology
- Traditional protective relay testing methods using physics-based models involve thousands of test cases. AI and ML based protection algorithms needs to be vetted through many more test cases to make sure the algorithms perform well for all anticipated fault and non-fault conditions.

- Taking shortcuts instead of going through rigorous training and testing and validation processes

In the last few decades, software has become part of various components and systems, and contributes directly to the success of their different functions and missions. It is an essential asset for current operation of IEDs, and with wide-scale digitalization it will become the critical central point of the future real-time analysis, prediction, prescription and decision-making of the power plant and substation. That is why, over the time, the size of software systems is continuously increasing, and they become more and more complex. So far, the reliability of software has rarely been considered in utility reliability studies or in the asset management. Based on the criticality incumbent to software, its risk is now requested to be integrated into the future studies.

There is one major difference between software and hardware engineering. The unreliability in software results in design faults which are mainly caused by human (intellectual) failures. On the other hand, hardware unreliability is based on random component failures. Here is a partial list of different characteristics between software and hardware[1]:

Wear-out	Software does not have energy related wear-out phase. Errors can occur without warning
Repairable system concept	Periodic restarts can help fix software problems
Time dependency and life cycle	Software reliability may not be a function of operational time
Environmental factors	Do not affect software reliability, except it might affect program inputs
Reliability prediction	Software reliability can not be predicted from any physical basis, since it depends completely on human factors in design
Redundancy	Can not improve software reliability if identical software components are used
Interfaces	Software interfaces are purely conceptual other than visual
Failure rate motivators	Usually not predictable from analyses of separate statements
Built with standard components	Well-understood and extensively-tested standard parts will help improve maintainability and reliability. But in software industry, we have not observed this trend. Code reuse has been around for some time but to a very limited extent. Strictly speaking there are no standard parts for software, except some standardized logic structures

Reliability theories have been developed over the years and used successfully for hardware systems. Previous work has been focused on extending the classical reliability theories

[1] "Software Reliability", Jiantao Pan. http://users.ece.cmu.edu/~koopman/des_s99/sw_reliability/

from hardware to software in such a way that we can establish software reliability framework consistently from the same viewpoints as hardware. But some major drawbacks were obvious :

- While hardware failures may occur independently (or approximately so), software failures do not happen independently.
- The interdependency of software failures is also very hard to describe in detail or to model precisely.
- Similar hardware systems are developed from similar specifications, and hardware failures, usually caused by hardware defects, are repeatable and predictable. Software systems are typically one-of-the-kind.

Thus, even though it is very difficult to apply at first the usual tools such as FMEA (Failure Mode and Effect Analysis) and FMECA (Failure Mode and Effect Criticality Analysis) for software reliability, it is on the other hand crucial that we put a lot of effort right away to include the software reliability analysis in those studies since the software is omnipresent in most of the hardware used in critical utility components and systems.

7.2 Challenges

A crucial aspect for the AI/ML based methods to succeed is the availability of reliable field data. It is very hard to find large sets of field data, especially for protection, since these data need to faithfully capture all disturbances and faults, so both desired dependability and security of protection could be achieved .

For example, for fault location application, the field data need to capture all kinds of faults, with different fault types, fault locations, fault inception angles and fault resistance; for protection applications, these data need to faithfully capture not only extensive internal faults but also all disturbances (external faults or other system transients). Take transmission lines with voltage level higher than 220kV with the State Grid Corporation of China (SCGG) as examples. Through year 2020, there were around 2000 transmission line faults, and the overall length of transmission lines is around 620,000 kilometers [105]. That is to say, on average there is only 1 fault per year for a transmission line with the length of 310 km. One can observe that the field data during faults are extremely sparse both spatially and temporally. There are several possible ways to use the very limited amount of field data, as shown in Figure 7-1. Details are elaborated below.

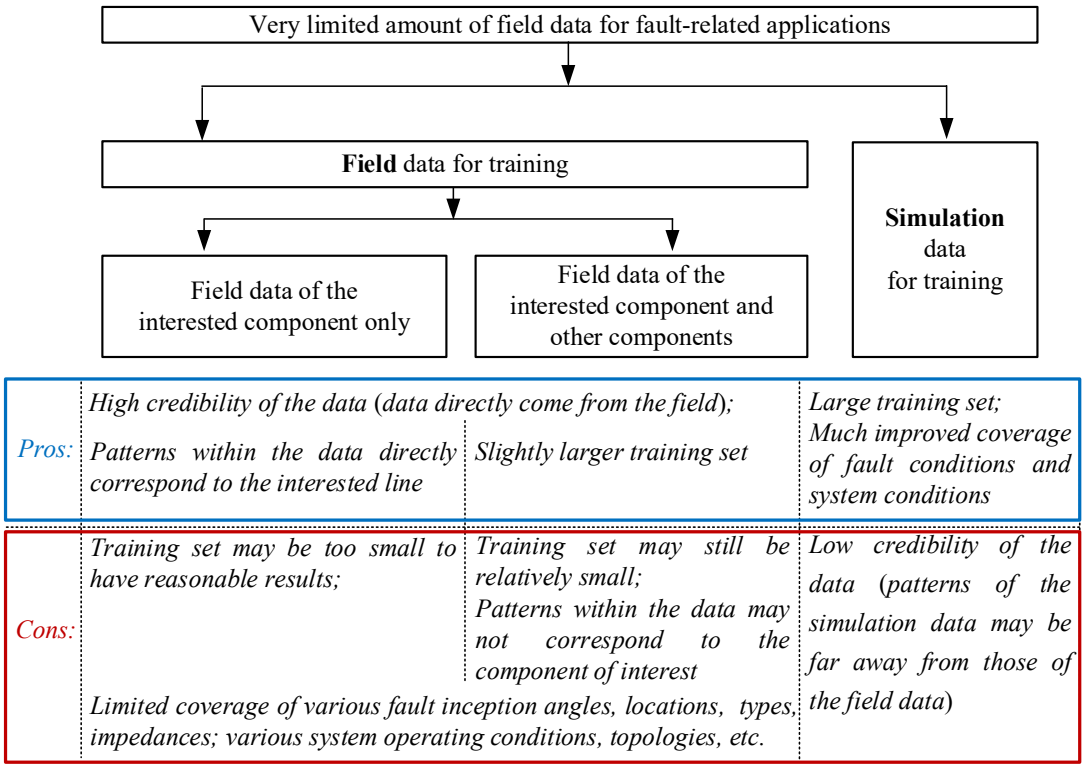


Figure 7-1: Pros and Cons of AI/ML based methods for fault-related applications

First, if the field data of only the interested component are adopted for training, since the training set may be too small with insufficient coverage of fault conditions (such as fault types, locations, inception angles, and resistances), it is unlikely to have acceptable results. In addition, learning may be required every time there are changes in topology or other operating conditions of the system, which may take a protective relay out of service and leave the network unprotected or cause considerable delays of the fault location/classification procedure.

Second, besides the field data of the interested component, if the field data of other components are also adopted to enlarge the training set, the patterns within the data may not correspond to the component of interest only. In this case, portability of data generated in one component/network to train for other components/networks may be questionable. Scalability of such solutions have been shown to be poor in published literature. In addition, if an AI/ML based method uses data of multiple resolutions to create "richer" patterns, such data would come from different types of sensors, each sensor potentially having unique errors. This also brings challenges to the performances of AI/ML based methods.

Finally, to solve the issues of insufficient coverage and small training set, large training sets can be generated using the simulation software. Note that the generated dataset at least needs to be "complete" in a sense that it has considered all aspects of fault conditions and system conditions; otherwise, overfitting issues may occur [106]. Even so, the credibility

of the generated training data is still questionable: how to validate that the patterns of the training data generated by the simulation software are consistent with those of the field data? It is observed that faults often tend to have different signatures than those gleaned from system-analytics or simulations. In fact, physics-based relays may perform better in such cases, and even if they fail, they fail "predictably"; AI/ML based methods tend to perform poorly when encountered with signatures not seen while training. It is also easier to understand failures of physics-based methods than data-driven, learning based methods, making the physics-based methods easier to correct. Nevertheless, the physics-based methods may also suffer from modeling inaccuracy since the calibration of system models in practical power systems are also challenging [107]. The bottom line is, before applying an AI/ML based method into practical power systems, one would need to evaluate the performance of the method using the field data, regardless of whether the AI/ML based method is trained using simulation or field data.

Interpretation of reliability of AI/ML based methods also needs to be considered rationally. Even with 99.9% success rate of an AI/ML based scheme on a protected network, dozens of daily misoperations or nonoperations would result if the technology is widely deployed across a major grid (thousands of networks). This goes to show that currently deployed protection schemes, though they experience failures from time to time, have an extremely high-performance baseline.

There are, however, problems in power system protection that do not yield to clear physics-based modeling. Such problems would benefit from AI/ML based solutions if data are available. AI/ML community has already recognized the issues with opaque models and limited availability of field data. This has prompted research into physics-aware AI/ML, which is expected to capture the best of both approaches in creating better solutions than those that exist today.

Sometimes, staged-fault testing is done to collect field data to verify newer fault detection techniques for various surfaces [108]. However, staged-fault testing is expensive and time consuming and can impact system security due to failure of equipment/system during such testing. Many utilities are moving away from such testing.

7.3 Criteria for Accepting AI/ML Applications in Field

In general, it is necessary to do a study/proof of concept before accepting an AI/ML application in the actual utility power system. The study typically contains:

- The impact of a failure of the device, system or any element that is being replaced or improved.
- Cost of implementation, operation and benefit gain
- Schedule from proof of concept to implementation date
- Feedback/review plan for adjustment of deployed system (maintenance cost)
- Personnel training plan to use the system.
- List of departments impacted and how they need to adapt to the new system
- Change management, plan for regularly review/audit documentation and results of the AI/ML decision.

General recommendations to address the definition of a process for accepting AI/ML applications in the field are as follows.

Main criteria points are planning, effective communication, progressive implementation, process changing, main risk scenarios testing, redundancy in case of failure.

7.3.1 Planning

Implementing new technologies requires a good strategy because of two aspects, i.e., human nature has a resistance for changes and changing technology has risks, that need to be properly analyzed and controlled. Stay high level during this stage and try not to get down to the detail.

In the planning stage there need to be:

1. A problem to be solved; otherwise, it will be a very difficult path to move forward in the AI/ML.
2. A benefit in savings of time and money.
3. An explanation of how AI/ML will solve this problem
4. A strategy on how to implement these successfully.

There needs to be a commitment from the decision maker that this is worth investing. Back to the planning stage, the planning document that is convincing and wholesome is critical.

7.3.2 Effective Communication

Keeping people informed about the AI/ML implementation process, steps, benefits, testing stages and risks and their managements is fundamental. Regular communication needs to take place to provide updates on implementation progress. Users could establish an open and direct communication to developers when finding and resolving issues. Users could also communicate lessons learned during each stage.

7.3.3 Progressive Implementation

When starting with the operation of a new AI/ML solution, device, system, start with no critical points, elements or substation, do not compromise the power system security.

Aim small and be nimble. Start small, review the results, update/improve and repeat. Slowly increase the implementation plan or area. This method provides a place to fail safely and allow the team to come back up and try again really quick. The key is to be honest when things failed, to change and get going again.

A continuous or even periodic comparison against a non-AI/ML solution needs to happen to update the learning and training process.

7.3.4 Process Changing

Taking the right time to change process and to update team training is important to avoid misunderstandings when using a new AI/ML solution. Provide the right documentation to

understand the solution. People can use AI/ML better when they have a better understanding about how the black box works.

Make sure management is included in the new process change and provide reports on how the new AI/ML solution is improving and meeting the goals/intended purpose.

Provide forum to discuss changes and open communication to developer/maintainer of the system if there are any problem. This has to be treated as continuous improvement.

Business process may change because of this AI/ML implementation and not once but a few times throughout the life of the AI/ML system.

7.3.5 Main Risk Scenarios Testing

Main risk scenarios testing of the AI/ML solution needs to be carried out with minimum two critical scenarios using real data. Depending on the complexity of applications, many scenarios may need to be considered during testing phase. Testing stage is fundamental for any solution or idea for changing conventional methods.

Future regulatory (for example, NERC PRC standards) compliance testing requirements of AI/ML systems need to be taken into account.

7.3.6 Redundancy in Case of Failure

Having a plan B when applying a new AI/ML solution in critical points of the power system where security could be compromised. Redundancy is desired. The following are examples of redundancy that can be included:

- Computer systems
- Network
- Sensors
- Backups/Restoration

8 CONCLUSIONS AND RECOMMENDATIONS

This report demonstrates that the practical application of limited AI/ML technology for power system protection and control has already started. The two practical use-case examples discussed in this report are clear evidence that the technology could be applied successfully to solve practical power system protection and control problems. The other examples highlighted also illustrate that there are many more areas that AI/ML technology has a great potential to help solve some of the power system protection and control challenges.

It is also very clear that practical and widespread application and deployment of this technology is still at a very early stage. Despite a large number of research and development efforts that have been reported to date, many of such efforts are still a long way from being applied mainstream. The development, implementation, testing, and user acceptance of

AI/ML technology-based applications still have some way to go before achieving deployment maturity and wide-spread acceptance by utilities and practicing engineers.

It is the hope of the authors that this report will serve to help accelerate the adoption of the AI/ML technology for practical power system protection and control applications. Given the large number of on-going R&D efforts, the body of the knowledge in this area is expected to grow and expand rapidly. Being an emergent area, it is not possible to attempt addressing details of all aspects in this report within a limited timeframe, but care has been taken so that this report will still be relevant in a few years after the body of the knowledge has expanded based on more widespread deployments globally.

Therefore, this report may become a living document that needs updating on a regular basis (e.g., every 3-4 years) to help incorporate new knowledges/insights and the latest progress in the practical application of the AI/ML technology in power system protection and control.

9 APPENDIX A – BIBLIOGRAPHY

Books

Jennie Si; Andrew G. Barto; Warren Buckler Powell; Don Wunsch, *Handbook of Learning and Approximate Dynamic Programming* , IEEE, 2004,

Sutton, R. & Barto, A. *Reinforcement Learning: An Introduction* (MIT Press, 1998)

Sutton, R.S. *Learning to predict by the methods of temporal differences*. Mach Learn 3, 9–44 (1988). https://doi.org/10.1007/BF00115009

Papers

[1] A. Mosavi, M. Salimi, S. Ardabili, T. Rabczuk, S. Shamshirband and A. Varkonyi-Koczy, "State of the Art of Machine Learning Models in Energy Systems, a Systematic Review," *Energies,* vol. 12, 2019.

[2] L. Wu, G. Kaiser, C. Rudin, D. Waltz, R. Anderson, A. Boulanger, H. Dutta and M. Pooleery, "Evaluating Machine Learning for Improving Power Grid Reliability," in *ICML Workshop on Machine Learning for Global Challenges*, 2011.

[3] C. B. Jones, A. Summers and M. J. Reno, "Machine Learning Embedded in Distribution Network Relays to Classify and Locate Faults," in *IEEE Innovative Smart Grid Technologies (ISGT)*, 2021.

[4] S. Hossain-McKenzie, E. C. Piesciorovsky, M. J. Reno and J. C. Hambrick, "Microgrid Fault Location: Challenges and Solutions," Sandia National Laboratories, SAND2018-6745, 2018.

[5] A. C. Adewole, R. Tzoneva and S. Behardien, "Distribution Network Fault Section Identification and Fault Location Using Wavelet Entropy and Neural Networks," *Applied Soft Computing,* vol. 46, pp. 296-306, 2016.

[6] A. Prasad, J. B. Edward and K. Ravi, "A Review of Fault Classification Methodologies in Power Transmission Systems: Part-I," *Journal of Electrical Systems and Informaton Technology,* vol. 5, no. 1, pp. 48-60, 2018.

[7] Y. Wang, M. Lui and Z. Bao, "Deep Learning Neural Network for Power System Fault Diagnosis," in *Proceedings of the 35th Chinese Control Conference*, 2016.

[8] S. K. Saha, P. Das and A. K. Chakrabothy, "An ANN based Relay Design for Identification Faults of 400kv High Voltage AC Transmission Line," *International Journal of Computer Applications,* vol. 46, no. 20, 2012.

[9] V. Malathi, N. S. Marimuthu, S. Baskar and K. Ramar, "Application of extreme learning machine for series compensated transmission line protection," *Engineering Applications of Artificial Intelligence,* vol. 24, no. 5, pp. 880-887, 2011.

[10] F. Wilches-Bernal, A. Bidram, M. J. Reno, J. Hernandez-Alvidrez, P. Barba, B. Reimer, R. Montoya, C. Carr and O. Lavrova, "A Survey of Traveling Wave Protection Schemes in Electric Power Systems," *IEEE Access,* 2021.

[11] R. Montoya, B. Poudel, A. Bidram and M. J. Reno, "DC Microgrid Fault Detection Using Multiresolution Analysis of Traveling Waves," *International Journal of Electric Power & Energy Systems,* 2022.

[12] M. Jimenez-Aparicio, M. J. Reno, P. Barba and A. Bidram, "Multi-Resolution Analysis Algorithm for Fast Fault Classification and Location in Distribution Systems," in *IEEE International Conference on Smart Energy Grid Engineering*, 2021.

[13] M. J. Aparicio, S. Grijalva and M. J. Reno, "Fast Fault Location Method for a Distribution System with High Penetration of PV," in *Hawaii International Conference on System Sciences (HICSS)*, 2021.

[14] S. S. Venkata, M. J. Reno, W. Bower, S. Manson, J. Reilly and G. W. S. Jr., "Microgrid Protection: Advancing the State of the Art," Sandia National Laboratories, SAND2019-3167, 2019.

[15] Z. Chen, L. Wu, S. Cheng, P. Lin, Y. Wu and W. Lin, "Intelligent fault diagnosis of photovoltaic arrays based on optimized kernel extreme learning machine and I-V characteristics," *Applied Energy,* vol. 204, pp. 912-931, 2017.

[16] M. Awaad, S. Mekhamer and A. Y. Abdelaziz, "Design of an adaptive overcurrent protection scheme for microgrids," *International journal of engineering science and technology,* vol. 10, no. 1, pp. 1-12, 2018.

[17] Q. Cui and S. Li, "A Microgrid Protection Scheme with Conventional Relay Measurements," in *2018 IEEE Power & Energy Society General Meeting (PESGM)*, 2018.

[18] A. Iqbal and . Pooja, "Intrusion Detection in Smart Grid Using Machine Learning Approach," *Journal of Computational and Theoretical Nanoscience,* vol. 16, no. 9, pp. 3808-3816, 2019.

[19] H. Lin, J. M. Guerrero, C. Jia, Z.-h. Tan, J. C. Vasquez and C. Liu, "Adaptive overcurrent protection for microgrids in extensive distribution systems," in *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, 2016.

[20] M. Manohar, E. Koley and S. Ghosh, "Microgrid protection under wind speed intermittency using extreme learning machine," *Computers & Electrical Engineering,* vol. 72, pp. 369-382, 2018.

[21] M. Mishra and P. K. Rout, "A comprehensive micro-grid fault protection scheme based on S-transform and machine learning techniques," *International Journal of Advanced Mechatronic Systems,* vol. 7, no. 5, p. 274, 2017.

[22] N. E. Naily, S. M. Saad, J. Wafi, A. Elhaffar and N. Husseinzadch, "Adaptive Overcurrent Protection to Mitigate High Penetration of Distributed Generation in Weak Distribution Systems," in *2017 9th IEEE-GCC Conference and Exhibition (GCCCE)*, 2017.

[23] W. J. Tang and H.-T. Yang, "Data Mining and Neural Networks Based Self-Adaptive Protection Strategies for Distribution Systems with DGs and FCLs," *Energies,* vol. 11, no. 2, p. 426, 2018.

[24] M. Mishra and P. K. Rout, "Detection and classification of micro-grid faults based on HHT and machine learning techniques," *IET Generation, Transmission, & Distribution,* vol. 12, no. 2, pp. 388-397, 2018.

[25] E. Casagrande, W. L. Woon, H. H. Zeineldin and D. Svetinovic, "A Differential Sequence Component Protection Scheme for Microgrids With Inverter-Based Distributed Generators," *IEEE Transactions on Smart Grid,* vol. 5, no. 1, pp. 29-37, 2014.

[26] S. Afrasiabi, M. Afrasiabi, B. Parang and M. Mohammadi, "Integration of Accelerated Deep Neural Network into Power Transformer Differential Protection," *IEEE Transactions on Industrial Informatics,* pp. 1-1, 2019.

[27] S. Das and B. K. Panigrahi, "Online intelligent technique for preventing relay maloperation under stressed conditions," in *2017 19th International Conference on Intelligent System Application to Power Systems (ISAP)*, 2017.

[28] R. Dubey, S. Samantaray and B. Panigrahi, "An extreme learning machine based fast and accurate adaptive distance relaying scheme," *International Journal of Electrical Power & Energy Systems,* vol. 73, pp. 1002-1014, 2015.

[29] Y. Feng, B. Duan, C. Tan and Z. Yao, "A settings tracking and providing scheme for differential protection based on machine learning," in *2015 IEEE Innovative Smart Grid Technologies - Asia (ISGT ASIA)*, 2015.

[30] M. Tasdighi and M. Kezunovic, "Preventing transmission distance relays maloperation under unintended bulk DG tripping using SVM-based approach," *Electric Power Systems Research,* vol. 142, pp. 258-267, 2017.

[31] S. Zhang, Y. Wang, M. Liu and Z. Bao, "Data-Based Line Trip Fault Prediction in Power Systems Using LSTM Networks and SVM," *IEEE Access,* vol. 6, pp. 7675-7686, 2018.

[32] R. C. Matthews, T. R. Patel, A. Summers, M. J. Reno and S. Hossain-McKenzie, "Per-Phase and 3-Phase Optimal Coordination of Directional Overcurrent Relays Using Genetic Algorithm," *Energies,* 2021.

[33] T. Patel, S. Brahma, J. Hernandez-Alvidrez and M. J. Reno, "Adaptive Protection Scheme for a Real-World Microgrid with 100% Inverter-Based Resources," in *IEEE Kansas Power and Energy Conference (KPEC)*, 2020.

[34] H. Lin, K. Sun, Z. Tan, C. Liu, J. M. Guerrero and J. C. Vasquez, "Adaptive protection combined with machine learning for microgrids," *IET Generation, Transmission, & Distribution,* vol. 13, no. 6, pp. 770-779, 2019.

[35] Y. Zhang, M. D. Ilíc and O. K. Tonguz, "Mitigating Blackouts via Smart Relays: A Machine Learning Approach," in *Mitigating Blackouts via Smart Relays: A Machine Learning Approach*, 2011.

[36] E. Bernabeu, J. Thorp and V. Centeno, "Methodology for a Security/Dependability Adaptive Protection Scheme Based on Data Mining," *IEEE Transactions on Power Delivery,* vol. 27, no. 1, 2012.

[37] G. N. Korres and P. J. Katsikas, "Identification of circuit breaker statuses in WLS state estimator," *IEEE Transactions on Power Systems,* vol. 17, no. 3, pp. 818-825, 2002.

[38] C. Francis, R. D. Trevizan, M. J. Reno and V. Rao, "Topology Identification of Power Distribution Systems Using Time Series of Voltage Measurements," in *IEEE Power and Energy Conference at Illinois (PECI)*, 2021.

[39] B. Poudel, D. R. Garcia, A. Bidram, M. J. Reno and A. Summers, "Circuit Topology Estimation in an Adaptive Protection System," in *IEEE North American Power Symposium (NAPS)*, 2021.

[40] A. Rajendra-Kurup, M. Martinez-Ramon, A. Summers, A. Bidram and M. J. Reno, "Deep learning based circuit topology estimation and fault classification in distribution systems," in *2021*, IEEE PES ISGT Europe.

[41] V. Kekatos and G. B. Giannakis, "Joint power system state estimation and breaker status identification," in *2012 North American Power Symposium (NAPS)*, Champaign, 2012.

[42] R. Dobbe, W. Westering, S. Liu, D. Arnold, D. Callaway and C. Tomlinar, "Forecasting-based state estimation for three-phase distribution systems with limited sensing," in *arXiv preprint arXiv:1806.06024*, 2019.

[43] J. Xie, A. P. S. Meliopoulos and B. Xie, "Transmission line fault classification based on dynamic state estimation and support vector machine," in *2018 North American Power Symposium (NAPS)*, Fargo, 2018.

[44] A. S. Zamzam, X. Fu and N. D. Sidiropoulos, "Data-driven learning-based optimization for distribution system state estimation," *IEEE Transactions on Power Systems,* To be published.

[45] O. Simeone, "A very brief introduction to machine learning with applications to communication systems," *IEEE Transactions on Cognitive Communications and Networking,* vol. 4, no. 4, pp. 648-664, 2018.

[46] T. Skripcak and P. Tanuska, "Utilisation of on-line machine learning for SCADA system alarms forecasting," in *2013 Science and Information Conference*, London, 2013.

[47] M. Teixeira, T. Salman, M. Zolanvari, R. Jain, N. Meskin and M. Samaka, "SCADA system testbed for cybersecurity research using machine learning approach," *Future Internet,* vol. 10, no. 76, 2018.

[48] L. A. Maglaras and J. Jiang, "A novel intrusion detection method based on OCSVM and K-means recursive clustering," *EAI Endorsed Transactions on Security and Safety,* vol. 2, no. 3, pp. 1-10, 2015.

[49] M. Zhou, Y. Wang, A. K. Srivastava, Y. Wu and P. Banerjee, "Ensemble-based algorithm for synchrophasor data anomaly detection," *IEEE Transactions on Smart Grid,* vol. 10, no. 3, pp. 2979-2988, 2019.

[50] A. Ahmed, V. V. G. Krishnan, S. A. Foroutan, M. Touhiduzzaman, A. Srivastava, Y. Wu, A. Hahn and S. Sindhu, "Cyber physical security analytics for anomalies in transmission protection systems," in *2018 IEEE Industry Applications Society Annual Meeting (IAS)*, Portland, 2018.

[51] W. Chen, S. Hsu and H. Shen, "Application of SVM and ANN for intrusion detection," *Computers & Operations Research,* vol. 32, pp. 2617-2634, 2005.

[52] S. Pan, T. Morris and U. Adhikari, "A specification-based intrusion detection framework for cyber-physical environment in electric power system," *International Journal of Network Security,* vol. 17, no. 2, pp. 174-188, 2015.

[53] U. Adhikari, T. H. Morris and S. Pan, "A cyber-physical power system test bed for intrusion detection systems," in *2014 IEEE PES General Meeting | Conference & Exposition*, 2014.

[54] V. Gurevich, Cyber and Electromagnetic Threats in Modern Relay Protection, 2014.

[55] T. Matsumoto, T. Kobayashi, S. Katayama, K. Fukushima and K. Sekiguchi, "Protection relay systems employing unconditionally secure authentication codes," in *2009 IEEE Bucharest PowerTech*, 2009.

[56] S. Rahman, H. R. Pota and J. Hossain, "Cyber vulnerabilities on agent-based smart grid protection system," in *2014 IEEE PES General Meeting | Conference & Exposition*, 2014.

[57] S. Hossain-McKenzie, M. J. Reno, R. Bent and A. Chavez, "Cybersecurity of Networked Microgrids: Challenges, Potential Solutions, and Future Directions," Sandia National Laboratories, SAND2020-13723, 2020.

[58] S. Hossain-McKenzie, M. J. Reno, J. Quiroz, P. Schulz, A. Summers and C. Carter, "Cyber Security Issues and Solutions for Protective Relaying and Local Monitoring," Sandia National Laboratories, SAND2018-0406, 2018.

[59] S. Ward, J. O'Brien, B. Beresh, G. Benmouyal, D. Holstein, J. T. Tengdin, K. Fodero, M. Simon, M. Carden, M. Yalla, T. Tibbals, V. Skendzic, S. Mix, R. Young, T. Sidhu, S. Klein, J. Weiss, A. Apostolov, D. Bui, S. Sciacca, C. Preuss, S. Hodder and G. Seifert, "Cyber Security Issues for Protective Relays; C1 Working Group Members of Power System Relaying Committee," 2007 IEEE Power Engineering Society General Meeting, Tampa, 2007.

[60] R. C. B. Hink, J. Beaver, M. Buckner, T. Morris, U. Adhikari and S. Pan, "Machine Learning for Power System Disturbance and cyber-attack

discrimination," in *2014 International Symposium on Resilient Control Systems (ISRCS)*, 2014.

[61] C.-W. Ten, J. Hong and C.-C. Liu, "Anomaly Detection for Cybersecurity of the Substations," *IEEE Transactions on Smart Grid,* vol. 2, no. 4, pp. 865-873, 2011.

[62] Y. Chen and B. Lou, "S2a: Secure smart household appliances," in *2nd ACM Conference Data Applicaiton Security Privacy*, San Antonio, 2012.

[63] R. Mitchell and I.-R. Chen, "Behavior-Rule Based Intrusion Detection Systems for Safety Critial Smart Grid Applications," *IEEE Transactions on Smart Grid,* vol. 2, no. 4, pp. 865-873, 2011.

[64] Y. Yang, McLaughlin, S. Sezer, T. Littler, B. Pranggono, P. Brogan and H. F. Wang, "Inrusion Detection System for Network Security in Synchrophasor Systems," in *IET Interational Conference*, 2013.

[65] H. Hadeli, R. Schierholz, M. Braendle and C. Tuduce, "Leveraging determinism in inudstrial control systems for advanced anomaly detection and relaible security configuration," *Emerging Technologies and Factory Automation,* pp. 22-25, 2009.

[66] J. Hong, R. F. Nuqui, A. Kondabathini, D. Ishchenko and A. Martin, "Cyber Attack Resilient Distance Protection and Circuit Breaker Control for Digital Substations," *IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS,* vol. 15, no. 7, 2019.

[67] J. Mulder, M. Schwartz, M. Berg, J. R. V. Houten, J. M. Urrea, M. A. King, A. A. Clements and J. Jacob, "WeaselBoard: Zero-Day Exploit Detection for Programmable Logic Controllers," Sandia National Laboratories, Albuquerque, 2013.

[68] "PFP Cybersecurity," 2019. [Online]. Available: https://www.pfpcyber.com/.

[69] Y. Huang, J. Tang, Y. Cheng, H. Li, K. A. Campbell and Z. Han, "Realtime detection of false data injection in smart grid networks: An adaptive cusum method and analysis," *IEEE Syst. J.,* vol. 10, no. 2, pp. 532-543, 2016.

[70] A. P. S. Meliopoulos, G. J. Cokkinides, P. Myrda, Y. Liu, R. Fan, L. Sun, R. Huang and Z. Tan, "Dynamic state estimation-based protection: Status and promise," *IEEE Transactions on Power Delivery,* vol. 32, no. 1, pp. 320-330, 2017.

[71] H. Lin, K. Sun, Z.-H. Tan, C. Liu, J. M. Guerrero and J. C. Vasquez, "Adaptive Protection Combined with Machine Learning for Microgrids," *IET Generation Transmission & Distribution,* 2019.

[72] Turing, A. "Computing Machinery and Intelligence" Mind, October 1950, LIX (236): 433–460

[73] Silva I, Spatti D. H., Flauzino R., et al, Artificial Neural Networks, a Practical Course, Springer 2017.

[74] Yang, Xin-She, et al, Artificial Intelligence, Evolutionary Computing and Metaheuristics, in the Footsteps of Alan Turing, Springer, 2013.

[75] Cao L. Metasynthetic Computing and Engineering of Complex Systems, Advanced Information and Knowledge Processing, Springer, 2015.

[76] Jones, M. T, Artificial Intelligence A Systems Approach, Computer Science Series, Jones & Bartlett Learning, 2009.

[77] Kumar A., Great Mind Maps for Learning Machine Learning, Data Analytics, Data Science, Machine Learning AI, Dec 28, 2020. Accessed on: May 20, 2021. [Online]. Available: https://vitalflux.com/great-mind-maps-for-learning-machine-learning/

[78] Grosan C, Abraham A., Intelligent Systems, A Modern Approach, Springer, Vol 17, 2011

[79] Y. Bengio, I. Goodfellow and A. Courville, Deep Learning, London: MIT Press, 2016.

[80] Y. Bengio, A. Courville and P. Vincent, "Representation Learning: A Review and New Perspectives," 2014.

[81] J. Schmidhuber, "Deep Learning in Neural Networks: An Overview," Lugano, 2014.

[82] H. -. Jacobsen, "A generic architecture for hybrid intelligent systems," 1998 IEEE International Conference on Fuzzy Systems Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98CH36228), 1998, pp. 709-714 vol.1, doi: 10.1109/FUZZY.1998.687575.

[83] Jeffrey Wischkaemper, Sukumar Brahma, “Machine Learning and Power System Protection [Viewpoint]”, *IEEE Electrification Magazine*, March 2021, pp. 108-112.

[84] Jörg Blumschein, Yilmaz Yelgin, Andrea Ludwig, “Adaptive Autoreclosure to Increase System Stability and Reduce Stress to Circuit Breakers”, presented at the 70th Annual Conference for Protective Relay Engineers, April 3-6, 2017

[85] Piñeros J. “Intelligent Model to Determine and Verify Automatically Distance Protection Settings”, Research work for M.Sc. Degree, Universidad de Antioquia, Medellín Colombia, 2016.

[86] Peter B. SnowAlexander P. ApostolovJefferson D. Bronfeld, US Patent 5,537,327, “Method and apparatus for detecting high-impedance faults in electrical power system”, July 16, 1996

[87] Peter B. SnowAlexander P. ApostolovJefferson D. Bronfeld, US Patent 5,734,575 “Method and apparatus for detecting high-impedance faults in electrical power systems”, March 31, 1998

[88] Nikos Hatziargyriou et al., “Definition and Classification of Power System Stability – Revisited & Extended”, IEEE Transactions on Power Systems, Vol. 36, No. 4, July 2021.

[89] K.. Dharmapala, A. Rajapakse, K. Narendra and Y. Zhang, "Machine Learning Based Real-Time Monitoring of Long-Term Voltage Stability Using Voltage Stability Indices," in IEEE Access, vol. 8, pp. 222544-222555, 2020.

[90] Power System Relaying Committee Report, “Transmission relay system performance comparison,” 2005. [Online]. Available: http://www.pes-psrc.org/Reports/I17ReportRev3.zip

[91] D. C. E. de la Garza, Masters Thesis - Hidden Failures in Protection Systems and its Impact on Power System Wide-area Disturbances. Virginia Polytechnic Institute and State University, 2000.

[92] O. P. Dahal, H. Cao, S. Brahma, and R. Kavasseri, “Evaluating performance of classifiers for supervisory protection using disturbance data from phasor measurement units,” in 2014 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe), Istanbul, Oct 2014.

[93] S. Brahma, R. Kavasseri, Huiping Cao, N. R. Chaudhuri, T. Alexopoulos, and Y. Cui, “Real Time Identification of Dynamic Events in Power Systems using PMU data, and Potential Applications – Models, Promises, and Challenges”, IEEE Trans. Power Delivery – Special Issue on Innovative Research Concepts for Power Delivery Engineering, Vol. 32-1, pp. 294 – 301, Feb. 2017.

[94] O. Dahal, S. Brahma, and H. Cao, “Comprehensive clustering of disturbance events recorded by phasor measurement units,” IEEE Trans. Power Del., vol. 29, no. 3, pp. 1390–1397, Jun. 2014.

[95] M. Biswal, Y. Hao, P. Chen, S. Brahma, H. Cao, and P. DeLeon, “Signal features for classification of power system disturbances using PMU data,” Proc. Power Syst. Comput. Conf., Jun. 2016, pp. 1–7

[96] M. Brown, M. Biswal, S. Brahma, S. Ranade, and H. Cao, “Characterizing and quantifying noise in PMU data,” in Proc. IEEE Power Energy Soc. Gen. Meeting, Jul. 2016, pp. 1–6.

[97] Y. Yin, S. Alaeddini and Y. Fu, "Automatic Fault Analysis and Visualization of Digital Substation Event," 2020 IEEE Power & Energy Society General Meeting (PESGM), 2020, pp. 1-5, doi: 10.1109/PESGM41954.2020.9281882.

[98] M. J. Aparicio, M. Reno, P. Barba, and A. Bidram, “Multi-resolution analysis algorithm for fast fault classification and location in distribution systems,” International Conference on Smart Energy Grid Engineering (SEGE), 2021.

[99] R. Montoya, B. Poudel, A. Bidram, and M. J. Reno, “DC microgrid fault detection using multiresolution analysis of traveling waves,” International Journal of Electrical Power & Energy Systems, vol. 135, 107590, Feb. 2022.

[100] R. Yousefian, R. Bhattarai and S. Kamalasadan, "Transient Stability Enhancement of Power Grid With Integrated Wide Area Control of Wind Farms and Synchronous Generators," in IEEE Transactions on Power Systems, vol. 32, no. 6, pp. 4818-4831, Nov. 2017.

[101] R. Yousefian, A. Sahami and S. Kamalasadan, "Hybrid energy function based real-time optimal wide-area transient stability controller for power system stability," 2015 IEEE Industry Applications Society Annual Meeting, 2015.

[102] F. Gomez, A.D. Rajapakse, U.D. Annakkage and I.T Fernando, "Support Vector Machine-Based Algorithm for Post-Fault Transient Stability Status Prediction Using Synchronized Measurements," in IEEE Transactions on Power Systems, vol. 26, no. 3, pp. 1474-1493, August 2011.

[103] A. D. Rajapakse, F. Gomez, K. Nanayakkara, P. A. Crossley and V. V. Terzija, "Rotor Angle Instability Prediction Using Post-Disturbance Voltage Trajectories," in IEEE Transactions on Power Systems, vol. 25, no. 2, pp. 947-956, May 2010.

[104] Working Group on Centralized Substation Protection and Control, IEEE Power System Relaying Committee, “Advancements in Centralized Protection and Control within a Substation”, in Proc. IEEE Trans. Power Delivery, doi:10.1109/TPWRD.2016.2528958, Vol. 31, No. 4, pp. 1945-1952, August 2016.

[105] Annual National Reliability Report of China, National Energy Administration, 2020.

[106] Y. Xing, Y. Liu, and X. Zheng, "A Systematic Framework Presenting Impact of Dataset Completeness on Data Driven Approaches: A Transmission Line Fault Classification Study", IEEE Conference on Energy Internet and Energy System Integration (EI2), 2020.

[107] Y. Liu, et. al, "Dynamic State Estimation for Power System Control and Protection", in IEEE Trans. Power Syst., vol. 36, no. 6, pp. 5909-5921, Nov. 2021, doi: 10.1109/TPWRS.2021.3079395.

[108] M. Carpenter, et. al, "Staged-Fault Testing for High Impedance Fault Data Collection", 58th Annual Conference for Protective Relay Engineers, Texas A&M University, College Station, TX, April 5-7, 2005.

[109] IEEE Std 1782™-2014, "IEEE Guide for Collecting, Categorizing, and Utilizing Information Related to Electric Power Distribution Interruption Events"

[110] IEEE Std 1159™-2019, "IEEE Recommended Practice for Monitoring Electric Power Quality"

[111] IEEE Std C37.114TM-2014, "IEEE Guide for Determining Fault Location on AC Transmission and Distribution Lines"

[112] Scikit-learn: Machine Learning in Python, Pedregosa et al., Journal of Machine Learning Research 12, pp. 2825-2830, 2011. Available online at: https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

[113] R. Das and S. A. Kunsman, "A Novel Approach for Ground Fault Detection", 57th Annual Conference for Protective Relay Engineers, Texas A&M University, College Station, TX, March 30-April 1, 2004.

[114] M. R. Lyu, "Software Reliability Engineering: A Roadmap," Future of Software Engineering (FOSE '07), 2007, pp. 153-170, doi: 10.1109/FOSE.2007.24.