A Comparison of Deterministic Automation and Machine Learning-Based Approaches for Protective Relay Settings: A Formal Specification Perspective

Reza Teshnizi, PhD
SynchroGrid
Bryan, TX USA
reza.teshnizi@synchrogrid.com

Luke Hankins, P.E.
SynchroGrid
Bryan, TX USA
lukeh@synchrogrid.com

***Abstract*—Protective relay settings are critical to the safe and reliable operation of the electric power grid, yet the process of developing relay settings remains a highly manual and time-intensive task. The increasing complexity and size of the power grid has naturally led to a demand for software-assisted tools that can improve accuracy and reduce the time required to develop relay settings. This paper presents a formal specification framework in which a utility's protection philosophy is expressed as a deterministic function of fault study outputs and equipment data. We compare two approaches for computing relay settings: a) evaluating the protection philosophy directly through automated fault studies and prescribed equations, and b) training machine learning models to approximate setting values from a set of system and equipment features. We present a case study using a synthesized realistic grid model with roughly 4600 lines and instantaneous overcurrent pickup elements protecting each line. We use the relay setting samples obtained from this grid to train four machine learning architectures across four feature configurations that systematically include or exclude the fault current and mutual coupling parameters, and evaluate their performance on three data splits: random hold-out, mutual coupling hold-out, and 345 kV voltage class hold-out. We show that the best model achieves 1.2% mean absolute percentage error when the fault current is provided as input but degrades to 35% when it is removed. We conclude that deterministic automation provides the most reliable, accurate, and standards-compliant method for computing protective relay settings, with machine learning better suited for problems involving optimization of free parameters rather than a replacement for core protection engineering calculations.**

***Index Terms*—Protective Relays, Formal Specifications, Relay Settings, Power System Protection, Automation, Deterministic Computation, Machine Learning.**

I. Introduction

The electric grid is experiencing rapid growth, both in demand and complexity, driven by factors such as the integration of distributed energy resources, grid modernization, demand management, and increasing interconnections among regional systems [1]. According to NERC State of Reliability data compiled from 2016 through 2022, incorrect settings consistently account for approximately 30% of all protection system misoperations, with human error contributing to roughly 40% of misoperations overall [2]. This growth has led to a corresponding increase in the time and effort required to ensure the safe and reliable operation of the grid, particularly in the area of protective relay settings. To manage this complexity, utilities are increasingly leveraging software-assisted tools that improve accuracy and reduce the time required to develop relay settings [3], [4].

In recent years, rapid advances in computational hardware and the emergence of effective machine learning methods have extended the application of artificial intelligence techniques into engineering domains. At the same time, regulatory and research bodies have begun to evaluate where AI can be responsibly applied in safety-critical power system operations [5]–[7]. These developments have naturally prompted the question of whether relay settings computation, which traditionally has been performed through manual calculations and rule-based procedures, could similarly benefit from data-driven approaches.

This paper presents a formal framework in which a utility's protection philosophy is expressed as a deterministic function of fault currents, grid topology, and equipment data. Using this formalism, we perform a case study on a realistic transmission grid model to evaluate whether machine learning can effectively approximate the settings computed by the deterministic approach. The main contributions of this paper are:

- A formal framework that expresses relay settings computation as a deterministic function of a formally specified protection philosophy, fault study outputs, and equipment data.
- A comprehensive empirical comparison of deterministic computation and machine learning estimation of relay settings, spanning four ML architectures, four feature configurations, and three generalization-testing data splits.
- A practical taxonomy distinguishing where deterministic methods, optimization, and machine learning are each appropriately applied in protection engineering workflows.

The remainder of this paper is organized as follows. Section II reviews prior work and the emerging role of AI in protection engineering. Section III formally defines the relay settings computation problem. Section IV presents the experimental design and case study. Section V presents results.

Section VI discusses practical implications, and Section VII concludes.

II. Background

The growing interest in artificial intelligence across the energy sector has prompted regulatory and research bodies to issue guidance on its appropriate use in safety-critical applications. In 2024, the U.S. Department of Energy published an assessment of the potential benefits and risks of AI for critical energy infrastructure [6], and NERC released a white paper on the use of AI and machine learning in real-time system operations [5]. Both documents emphasize the need for trustworthiness, explainability, and careful risk management when deploying AI in grid operations. The National Renewable Energy Laboratory further explored this theme in a technical report on trustworthy AI in the control room [7]. These publications reflect a broader recognition that while AI offers substantial value in certain power system applications, its deployment in safety-critical contexts requires rigorous evaluation of where it genuinely adds capability versus where it introduces unnecessary risk.

Within protection engineering specifically, machine learning has demonstrated clear value in fault detection and classification, where the mapping from relay measurements to fault characteristics is a pattern recognition problem well suited to neural networks [8]–[10]. More recently, graph neural networks have been applied to fault diagnostics in distribution systems, leveraging the topological structure of the grid [11], [12]. Relay coordination has historically been addressed through mathematical optimization. Linear programming [13], particle swarm optimization [14], and mixed-integer linear programming [15] have been applied to find optimal time dial and curve settings subject to coordination constraints. These methods operate within the bounds of a formally defined protection philosophy, searching a feasible space but do not replace the underlying settings formulas.

Attempts to apply supervised learning directly to relay settings computation — i.e., predicting numeric setting values from system parameters — remain rare. Early work explored neural networks for adaptive distance relay boundaries [16], and support vector machines have been used to classify impedance trajectories into protection zones [17]. More recently, machine learning has been applied to select among pre-computed settings groups for microgrid applications [18]. However, none of these approaches replace the deterministic computation of settings from a formally specified philosophy with a learned approximation trained end-to-end on system parameters. This paper addresses that gap directly.

III. Problem Formulation

In this section, we formalize the concept of a protection philosophy as a specification and show how the specification is used to compute relay settings. We introduce the notation associated with a specification, describe how this specification is instantiated for a specific relay on a given grid, and provide an example for setting specification.

A. Formal Specification of Protection Philosophy

A protection philosophy for a single relay element type is a tuple $\phi = (\Sigma, \mathcal{V}, \mathcal{C}, f)$, wherein

- Σ is a fault specification written as a set of rules describing fault types expressed in terms of topological configurations,
- $\mathcal{V}$ is a set of variables associated with a given relay device and the line it protects,
- $\mathcal{C}$ is a set of numeric constants that apply across all relays of the same element type, and
- $f : \mathbb{R}^{|\Sigma|} \times \mathbb{R}^{|\mathcal{V}|} \times \mathbb{R}^{|\mathcal{C}|} \to \mathbb{R}$ is a function that takes as input a vector of fault currents, a vector of values for the variables, and the constants, and returns a setting value.

A utility's complete protection philosophy is the collection $\Phi = \{\phi_1, \phi_2, \ldots, \phi_n\}$, with one tuple for each relay element type. The tuples are evaluated for a given element in the system as described in the following subsection.

B. Deterministic Calculation of Relay Settings

When the philosophy is applied to a specific relay, r, in a grid, G, the specification is instantiated using the following steps. The grid is expressed as a graph $G = (V, E)$, where buses are vertices V and transmission lines are edges E. Each relay is associated with a bus at one end of an edge in E, and to collect the topological context for it graph traversal algorithms — e.g., depth-first search and breadth-first search [19] — can be used [2]. Using the topological context for a given relay, each fault specification in Σ is translated to a fault type and a set of locations the fault should be placed. For example, "SLG at remote bus" is translated to a single-line-to-ground fault at the remote bus connected to r via the line it protects. Next, the fault types and locations are used in a short-circuit solver to produce a vector of fault currents, denoted by $\mathbf{I}_r^{\Sigma} \in \mathbb{R}^{|\Sigma|}$. Each entry in $\mathbf{I}_r^{\Sigma}$ corresponds to one rule in Σ and records the maximum fault current observed across all locations prescribed by that rule. Finally, the variables in $\mathcal{V}$ are instantiated to their numeric values associated with relay r, producing the vector $\mathbf{v}_r \in \mathbb{R}^{|\mathcal{V}|}$. The setting for relay r is then calculated as

$$s_r = f\big(\mathbf{I}_r^{\Sigma},\ \mathbf{v}_r,\ \mathcal{C}\big)\,. \tag{1}$$

In the above formulation, the function f and the constants $\mathcal{C}$ are invariant across all relays of the same element type. In the next subsection, we illustrate the construction of the function f with a concrete example for a relay setting.

C. Relay Setting Example

Consider, as an example, the residual ground instantaneous overcurrent pickup element. The protection philosophy for this element is a tuple $\phi_{\text{IOC}} = (\Sigma_{\text{IOC}}, \mathcal{V}_{\text{IOC}}, \mathcal{C}_{\text{IOC}}, f_{\text{IOC}})$ where

$$\begin{aligned}\Sigma_{\text{IOC}} &= \left\{\begin{matrix}\text{SLG at remote bus,}\\ \text{SLG along mutual lines}\end{matrix}\right\},\\ \mathcal{V}_{\text{IOC}} &= \{\text{CTR}\},\\ \mathcal{C}_{\text{IOC}} &= \{\alpha\}, \text{and}\\ f_{\text{IOC}}(\mathbf{I}^{\Sigma_{\text{IOC}}}, \mathbf{v}, \mathcal{C}) &= \frac{\alpha \times \max(\mathbf{I}^{\Sigma_{\text{IOC}}})}{\text{CTR}}.\end{aligned}$$

Since $\mathcal{V}_{\text{IOC}}$ contains a single variable, $\mathbf{v}_r$ reduces to the scalar value of the current transformer ratio for relay r.

In the function f_{IOC}, α is a constant specified in the protection philosophy, $\mathbf{I}^{\Sigma_{\text{IOC}}}$ is the vector of the three times single-line-to-ground zero-sequence fault current as measured by a given relay, and CTR is the current transformer ratio of that relay. In a typical grid configuration, $\max(\mathbf{I}^{\Sigma_{\text{IOC}}})$, wherein $\max(\cdot)$ returns the largest element of the vector, equates to the current resulting from a remote bus single-line-to-ground fault. However, in the presence of mutual coupling effects, the maximum current may be the result of a fault at a location along one of the coupled lines. For this reason, Σ_{IOC} prescribes these two fault types. Given that f, $\mathcal{C}$, and Σ_{IOC} are all provided by the philosophy specification, the calculation of s_r involves no free parameters and every relay's setting is uniquely determined by the fault study outputs and equipment data. In Section IV, we use this element in the empirical evaluation of the machine learning approaches.

IV. Experimental Design

Given the mathematical formulation of a formally specified philosophy in the previous section, we now consider data-driven approaches that attempt to approximate the settings computation. All the code required for the experiments in this section is implemented in Python using the *scikit-learn* library [20].

A. Machine Learning Formulation

Data-driven approaches to replacing the function f, attempt to learn the approximation $\hat{f}_\theta$, parameterized by weights θ that are fitted to a training set of relay settings and their associated system features. That is, the model attempts to learn the mapping

$$\hat{s}_r = \hat{f}_\theta(\mathbf{x}_r) \tag{2}$$

where $\mathbf{x}_r$ is a feature vector describing relay r and its surrounding grid characteristics. In contrast to Equation (1), where s_r is computed from a known function f applied to fault study outputs and equipment data, the learned model $\hat{f}_\theta$ must infer this mapping from the features in $\mathbf{x}_r$, which may or may not include the fault current $\mathbf{I}_r^\Sigma$.

B. Data Collection

To evaluate the machine learning alternative described above, we use the ground instantaneous overcurrent pickup, as formalized in Section III-C. To construct a dataset for training a model, we synthesized large-scale transmission system representative of a major interconnection, comprising approximately 4600 transmission lines spanning three voltage levels: 69 kV, 138 kV, and 345 kV. To create a realistic variation in setting values, we placed a residual ground instantaneous overcurrent relay at each end of every line in the grid, as well as mutual coupling pairs to obtain zero-sequence fault currents that arise due to interactions in parallel transmission corridors. For each relay, the ground truth setting value was calculated by evaluating the philosophy tuple ϕ_{IOC} against the grid model, as described in Equation (1).

Feature	**Description**
Z_1 mag, ang	Positive-sequence line impedance
Z_0 mag, ang	Zero-sequence line impedance
Z_m mag, ang	Mutual coupling impedance
Z_{m0} mag, ang	Mutual segment zero-seq impedance
Z_{m1} mag, ang	Mutual segment pos-seq impedance
Mutual seg. length	Length of the coupled segment
Line length	Length of the protected line
CTR	Current transformer ratio
V_{nom}	Nominal voltage level
N_{term}	Number of line terminals
N_{par}	Number of parallel lines
Has mutual	Binary: mutual coupling present
I^{3I_0}	3I0 fault current

TABLE I: Feature Set for Machine Learning Models

Config	**Fault Current**	**Mutual Detail**	**Features**
C1	✓	✓	18
C2	✓		13
C3		✓	17
C4			12

TABLE II: Ablation Feature Configurations

C. Feature Set

For each relay r, a feature vector $\mathbf{x}_r$ is constructed from equipment and local system parameters. Table I summarizes the complete set of features considered in our case study. With the exception of the fault current, all other features are static per equipment, and therefore can be extracted directly from the grid model without performing simulation. However, the fault current, $I^{3I_0} = \max(\mathbf{I}_r^\Sigma)$ is the obtained via short-circuit simulation and is the same value used in the function f to calculate s_r.

To understand the importance of specific input features in model performance, we define four configurations that include or exclude two feature groups: the fault current and the mutual coupling segment information. Table II summarizes the configurations. Configuration C1 provides the model with all available features. This represents the best possible case in training a model. To verify whether information about the mutual coupling segments meaningfully impact training performance, Configuration C2 removes the mutual segment information. To test whether information about the characteristics of the mutual segments can compensate for the absence of the fault current, Configuration C3 removes only the fault current. Finally, Configuration C4 removes both the fault current and the mutual coupling segment information to examine whether a trained model can predict relay settings without any output from the short-circuit solver.

D. Model Architecture

For the case study, we evaluate four machine learning architectures including neural network and tree-based ensemble methods.

The first two models are multilayer perceptrons (MLPs). To study the effect of model capacity on performance, we evaluate two MLP architectures with different sizes: MLP-Small and MLP-Large. MLP-Small consists of three hidden layers with 64, 128, and 64 neurons respectively, and MLP-Large consists of four hidden layers with 256, 512, 256, and 128 neurons. Both networks use ReLU activation functions and are trained with the Adam optimizer using mean squared error loss, with early stopping on a 15% validation hold-out to prevent overfitting. For tree-based ensembles models we evaluate Random Forest and Gradient Boosting. Random Forest is trained with 500 decision trees with no maximum depth restriction and a minimum of two samples per leaf. Gradient Boosting is trained with 500 trees at a maximum depth of six and a learning rate of 0.05.

All features are standardized to zero mean and unit variance prior to training via a standard scaler fitted on the training set. The same scaler transformation is applied to the test set at evaluation time.

V. Results

This section describes the data splits used to evaluate generalization, present the results across feature configurations and architectures, and analyze the importance of individual features in model performance.

A. Data Splits

The models are evaluated using the following three data splits. The *random 80/20* split withholds 20% of randomly selected relays from training. We anticipate this split provides the most favorable training scenario, as the test set is drawn from the same distribution as the training set. The *mutual coupling hold-out* split excludes all relays on mutually coupled lines from training and uses them exclusively for testing. This provides insights into whether a model trained only on lines without mutually coupled pairs can generalize to the zero-sequence interactions induced by mutual coupling. Lastly, the *345 kV voltage class hold-out* split removes all 345 kV relays from training set. This verifies whether the model can perform well on a voltage class with different fault current magnitudes, impedances, and CT ratios.

B. Model Performance Evaluation

By construction, the deterministic approach arrives at the ground truth values on all splits, as the ground truth labels are themselves produced by evaluating the philosophy specification. This reflects the fact that when the philosophy prescribes an exact formula, the only correct output is the result of that formula. Therefore, we ask whether a learned model can match this output without access to the full computation pipeline.

Among the models described in Section IV-D, tree-based ensembles achieved the best performance, with negligible difference between the two. Table III presents the results of the Gradient Boosting as the model with the best performance. The results clearly show that removing the fault current from the features noticeably degrades the model's performance. When the fault current is included the mean absolute percentage error (MAPE) ranges between 1.2% and 5.6%. When the fault current is excluded the error is roughly 30 times worse, with MAPE ranging from 32.6% to 99.1%. This drop in performance confirms that the fault current is the main component in predicting the setting values and that no combination of system parameters allows the model to replicate the output of short-circuit simulation.

Conversely, adding the mutual coupling segment details (i.e.,C1 versus C2 and C3 versus C4) does not meaningfully impact the performance of the model. When mutually coupled lines are excluded from the training set, even the most favorable configuration shows 15.5% of predictions exceeding 5% error, indicating that the model cannot generalize to mutual coupling effects. On the 345 kV hold-out, configurations without the fault current perform poorly, with over 94% of predictions exceeding 10% error.

Fig. 1 and Fig. 2 show the predicted versus actual setting values for the Gradient Boosting model with and without the fault current, respectively. In the former, predictions cluster tightly around the diagonal with visible scatter on the mutual hold-out split. In the latter, predictions deviate substantially from the diagonal across all splits.

C. Model Architecture Comparison

To verify that the results are not an artifact of a particular model architecture, Table IV presents the MAPE and percentage of predictions exceeding 5% error for all four architectures on the random 80/20 split under Configurations C2 and C4.

With the fault current in the feature set, tree-based ensembles outperform the neural network architectures. Gradient Boosting achieves the lowest MAPE at 1.2%, followed by Random Forest at 1.7%, whereas the MLPs range from 3.9% to 4.8%. This advantage is consistent across all three data splits, as shown in Fig. 3. However, when the fault current is removed, all four architectures degrade to comparable error levels, with MAPE values between 32% and 37% and over 79% of predictions exceeding 5% error. The convergence in performance across fundamentally different model families confirms that the error is not a capacity or architectural limitation; it is indeed a consequence of the lacking capability to model the underlying physics. No model, regardless of its expressiveness, can estimate the fault current from static system parameters.

D. Feature Importance

To understand which features the models rely on when predicting the setting values, we compute *permutation importance*

Config	Random 80/20			Mutual Hold-out			345 kV Hold-out		
	MAPE	>5%	>10%	MAPE	>5%	>10%	MAPE	>5%	>10%
C1 (3I0 + mutual segment)	1.3%	4.5%	2.0%	5.6%	15.5%	15.1%	1.4%	6.8%	1.4%
C2 (3I0, no mutual segment)	1.2%	4.2%	2.3%	5.5%	15.5%	15.1%	1.2%	2.7%	0.3%
C3 (mutual segment, no 3I0)	34.5%	82.9%	69.6%	32.7%	87.8%	75.6%	99.1%	97.2%	94.0%
C4 (no 3I0, no mutual segment)	35.2%	82.1%	66.2%	32.6%	87.4%	75.1%	92.2%	96.5%	94.0%

TABLE III: Gradient Boosting Results Across Feature Configurations and Data Splits

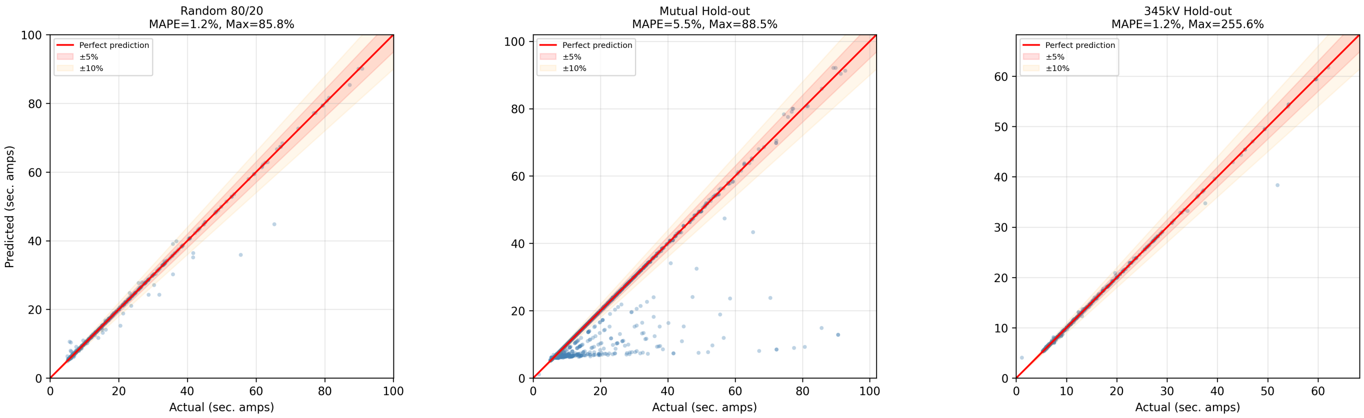

Fig. 1: Predicted vs. actual settings for Gradient Boosting with fault current (Configuration C2). The red and orange bands indicates a 5% and 10% error margin, respectively.

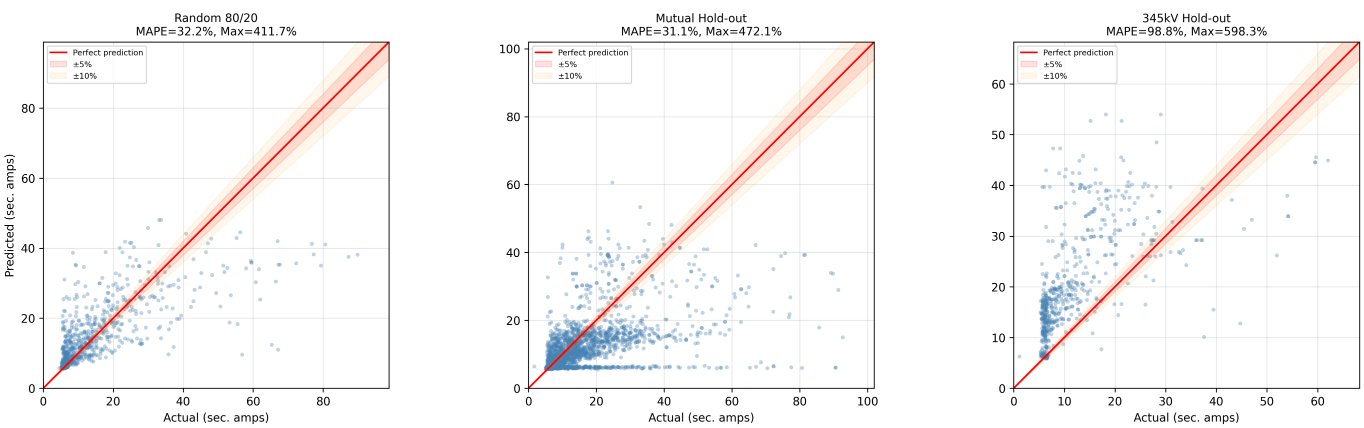

Fig. 2: Predicted vs. actual settings for Gradient Boosting without fault current (Configuration C4). The red and orange bands indicates a 5% and 10% error margin, respectively.

Model	C2 (with 3I0)		C4 (no 3I0)	
	MAPE	>5%	MAPE	>5%
MLP-Small	4.8%	25.7%	37.3%	84.0%
MLP-Large	3.9%	14.7%	32.4%	82.5%
Random Forest	1.7%	7.1%	32.2%	79.1%
Gradient Boosting	1.2%	4.2%	35.2%	82.1%

TABLE IV: Model Comparison on Random 80/20 Split

for all four architectures on the random 80/20 split under Configuration C1, where all features are available. Permutation importance measures the increase in prediction error when the values of a single feature are randomly shuffled to break its relationship with the target. Table V reports the permutation importance for the Gradient Boosting model, which achieved the best overall performance.

The results reveal a heavy reliance of predictive on just two features: the fault current with an importance of 16.59, and the CT ratio at 6.04. Together, these two features account for over 95% of the total permutation importance. All remaining features — including line impedances, mutual cou-

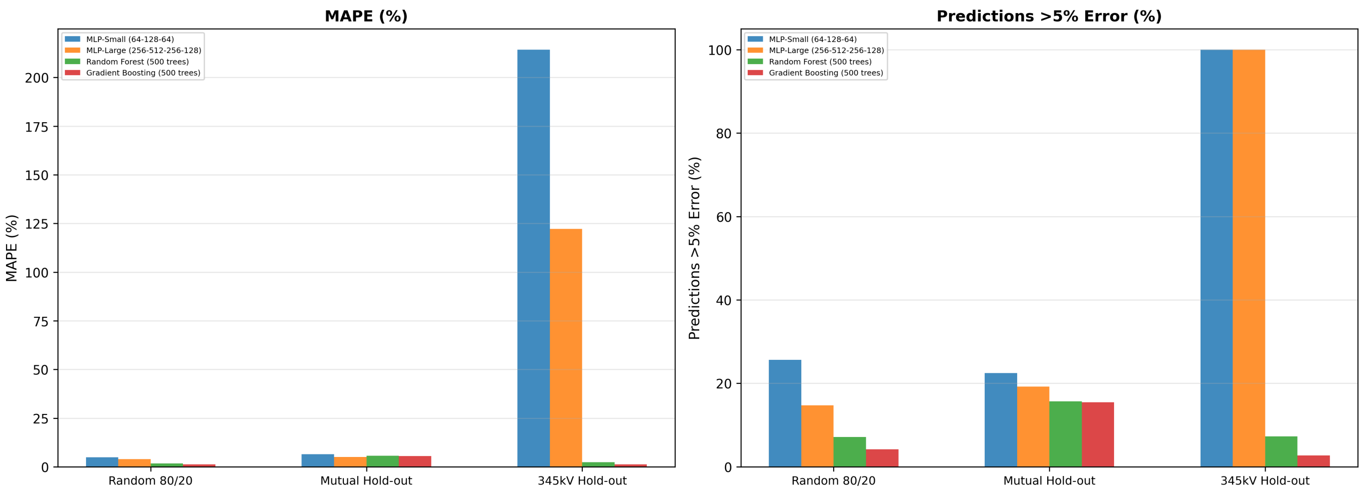

Fig. 3: MAPE and percentage of predictions exceeding 5% error across all four model architectures and three data splits (Configuration C2). Columns from left to right: MLP-Small, MLP-Large, Random Forest, Gradient Boosting.

pling parameters, voltage level, and topology descriptors — contribute negligibly. This pattern is consistent across all four architectures and directly reflects the structure of the function f_{IOC} used to compute the ground truth values, which is defined as $s = \alpha \times \max(\mathbf{I}^{\Sigma})/\text{CTR}$. The model is effectively learning the known relationship between the fault current, the CT ratio, and the setting value, that is α. The system parameters that describe the grid topology and line characteristics provide almost no additional predictive value, confirming that these features cannot substitute for the fault current when it is absent.

Feature	Importance
Fault Current	16.59
CTR	6.04
Mutual Z mag	0.12
Mutual seg. Z_0 mag	0.09
Mutual seg. Z_1 ang	0.06
Mutual seg. length	0.01
All other features	< 0.01

TABLE V: Permutation Importance — Gradient Boosting, C1

VI. Discussion

A. Why Machine Learning Fails at Settings Computation?

The results across all configurations, splits, and model architectures point to a single structural explanation. The computation of a relay setting has two distinct components: obtaining the fault current and evaluating the setting formula. The formula itself is trivial — a multiplication and a division. The fault current, however, is the output of a full network solution that accounts for the impedances of all interconnected elements, source contributions from every generator, and the topology of the grid under the specified fault condition. A machine learning model receiving only local equipment parameters — line impedances, CT ratios, voltage levels — has no mechanism to obtain this network solution. The fault current at a given relay depends not only on the protected line's impedance but on the parallel paths, source impedances, and interconnections throughout the surrounding grid. These quantities require a system-wide computation and cannot be inferred from the local feature vector.

Mutual coupling provides the clearest illustration of this limitation. As formalized in Section III, the maximum fault current measured by the instantaneous overcurrent element may occur at an intermediate point along a mutually coupled line, where induced zero-sequence currents increase the 3I0 magnitude above the remote bus fault current value. Identifying this location requires sweeping faults along all the coupled lines and comparing the resulting currents, requiring a search that the deterministic calculation process performs as part of its normal operation. A machine learning model sees only the static mutual impedance value in its feature vector and lacks sufficient information to simulate faults at varying locations or to discover that the fault current results from a specific point along a coupled segment.

This is not a limitation of model capacity or training data. It is a fundamental mismatch between the problem statement and the learning paradigm wherein the information needed to compute the setting is not contained in any fixed-length feature vector but in the solution of a system of network equations conditioned on a specific fault scenario.

B. Taxonomy of Protection Engineering Problems

The results presented in this work should not be interpreted as a general argument against the use of machine learning (or artificial intelligence in a broader sense) in protection engineering. Rather, they highlight the importance of matching the computational tool to the nature of the problem. We identify three distinct problem classes in protection engineering, each suited to a different approach.

Settings computation is the focus of this paper. When the protection philosophy is written as a formal specification via exact formulae, the inputs are available from a fault study and equipment data, then the computation is deterministic and produces the correct result with zero approximation error. There is no benefit to replacing this computation with a learned approximation.

Coordination optimization is a problem that arises when the philosophy permits a range of acceptable values. For instance, selecting a time dial setting or curve type that satisfies coordination time interval constraints with adjacent relays. This is a constrained optimization problem with genuine degrees of freedom, and methods such as mixed-integer linear programming, linear programming, and genetic algorithms are well suited to searching the feasible space for an optimal solution [21], [22]. Importantly, these methods operate within the constraints defined in the philosophy specification by selecting among compliant alternatives rather than replacing the settings formulas.

Fault detection and classification involves identifying the type and location of a fault from real-time relay measurements — waveforms, phasors, and sequence quantities. This is a pattern recognition problem where the governing equations are not known in closed form and where the mapping from measurements to fault characteristics must be inferred from data. Neural networks and other machine learning methods are genuinely well suited to this class of problems [8]–[10].

C. Practical Implications

The distinction between the three problem classes is clear. When the function is known the task is to simply evaluate it. When the feasible space must be searched, optimization methods are appropriate. When the mapping must be discovered, learning methods excel.

For settings computation, the practical advantages of the deterministic approach extend beyond accuracy. The process described in Section III is deployed at production scale at major utilities globally [3], [4]. When the protection philosophy is updated or the grid topology changes, the specification is re-evaluated and the affected relays are re-assessed by applying the revised philosophy to the current fault study data. Computed settings are fully traceable to the rules in the philosophy specification that produced them, the fault study that provided the current, and the equipment data that supplied the variables. This traceability is essential for compliance with regulatory standards — such as NERC PRC-027-1 [23] — which require documented, reviewable processes for the development and coordination of protection system settings.

In contrast, a machine learning approach would require retraining whenever the philosophy specification or system changes, relying on new labeled data that can only be generated by running the deterministic pipeline on the updated system — creating a redundancy in which the machine learning model depends on the very process it seeks to replace.

While a trained model can produce predictions faster than running a full short-circuit simulation for every relay, this speed gain is irrelevant in the face of the approximation errors demonstrated in this study. Moreover, the fault study must be run regardless for coordination verification, compliance documentation, and validation of protection schemes. Once the fault currents are available, evaluating the philosophy specification is computationally trivial.

VII. Conclusion

This paper presented a formal framework for specifying a utility's protection philosophy as a function of fault study outputs and equipment data, and provided an empirical comparison against machine learning approaches for computing protective relay settings. Through a case study on approximately 4,600 relays from a realistic transmission grid model, we evaluated four model architectures across four feature configurations and three data splits designed to test the models' capability in generalization. The best learned model achieves 1.2% MAPE when provided the fault current as input, but degrades to 35% MAPE when it is removed. Feature importance analysis confirms that the fault current and CT ratio account for over 95% of the model's predictive power, and that no combination of static system parameters can compensate for the absence of the simulated fault current. Adding detailed mutual coupling segment parameters provides no meaningful improvement under any configuration. We argued that deterministic automation provides the most reliable, accurate, and standards-compliant method for computing protective relay settings, while machine learning is better suited for problems involving optimization of free parameters or approximation of unknown functions, rather than a replacement for core protection engineering calculations. Future work will extend this comparison to additional relay element types, including distance and time overcurrent elements, and to broader test systems encompassing multiple grid models. We also plan to investigate the boundary between settings computation and coordination optimization more formally, and to evaluate new avenues for employing optimization and data-driven methods in adjacent areas such as setting calibration, misoperation analysis, and coordination.

References

[1] North American Electric Reliability Corporation, "2025 State of reliability: Assessment overview of 2024 bulk power system performance," NERC, Atlanta, GA, USA, Jun. 2025.

[2] N. Ganta, L. Hankins, and T. Goodwin, "Employing graph traversal techniques to simplify three and multi-terminal line applications," in *Proc. 77th Annu. Conf. Protective Relay Engineers (CPRE)*, College Station, TX, USA, 2024.

[3] J. Gurley, L. Hays, D. Ridley, Z. Austin, N. Thomas, L. Hankins, A. Borgnino, and J. Perez, "Interoperability of system protection software," in *Proc. 75th Annu. Conf. Protective Relay Engineers (CPRE)*, College Station, TX, USA, 2022, pp. 1–13.

[4] C. Navas, M. Khalid, L. Hankins, D. Ridley, Z. Austin, and R. Teshnizi, "Automating NERC PRC-027-1 compliance: A comprehensive approach to large-scale protection system coordination," in *Proc. 78th Annu. Conf. Protective Relay Engineers (CPRE)*, College Station, TX, USA, 2025.

[5] North American Electric Reliability Corporation (NERC), "Artificial intelligence and machine learning in real-time system operations: White paper – Revision 1," NERC Reliability and Security Technical Committee (RSTC), Nov. 2024.

[6] U.S. Department of Energy, Office of Cybersecurity, Energy Security, and Emergency Response (CESER), "Potential benefits and risks of artificial intelligence for critical energy infrastructure," Washington, DC, Apr. 2024.
[7] S. L. Choi, R. Jain, P. Emami, K. Wadsack, F. Ding, H. Sun, K. Gruchalla, J. Hong, H. Zhang, X. Zhu, and B. Kroposki, "eGridGPT: Trustworthy AI in the control room," National Renewable Energy Laboratory, Golden, CO, Tech. Rep. NREL/TP-5D00-87440, May 2024.
[8] T. Dalstein and B. Kulicke, "Neural network approach to fault classification for high speed protective relaying," *IEEE Trans. Power Del.*, vol. 10, no. 2, pp. 1002–1011, Apr. 1995.
[9] K. M. Silva, B. A. Souza, and N. S. D. Brito, "Fault detection and classification in transmission lines based on wavelet transform and ANN," *IEEE Trans. Power Del.*, vol. 21, no. 4, pp. 2058–2063, Oct. 2006.
[10] A. Yadav and Y. Dash, "An overview of transmission line protection by artificial neural network: Fault detection, fault classification, fault location, and fault direction discrimination," *Adv. Artif. Neural Syst.*, vol. 2014, Art. ID 230382, pp. 1–20, 2014.
[11] K. Chen, J. Hu, Y. Zhang, Z. Yu, and J. He, "Fault location in power distribution systems via deep graph convolutional networks," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 1, pp. 119–131, Jan. 2020.
[12] B. L. H. Nguyen, T. V. Vu, T.-T. Nguyen, M. Panwar, and R. Hovsapian, "Spatial-temporal recurrent graph neural networks for fault diagnostics in power distribution systems," *IEEE Access*, vol. 11, pp. 46039–46050, 2023.
[13] A. J. Urdaneta, R. Nadira, and L. G. Perez Jimenez, "Optimal coordination of directional overcurrent relays in interconnected power systems," *IEEE Trans. Power Del.*, vol. 3, no. 3, pp. 903–911, Jul. 1988.
[14] M. M. Mansour, S. F. Mekhamer, and N. E.-S. El-Kharbawe, "A modified particle swarm optimizer for the coordination of directional overcurrent relays," *IEEE Trans. Power Del.*, vol. 22, no. 3, pp. 1400–1410, Jul. 2007.
[15] S. T. P. Srinivas and K. Shanti Swarup, "A new mixed integer linear programming formulation for protection relay coordination using disjunctive inequalities," *IEEE Power Energy Technol. Syst. J.*, vol. 6, no. 2, pp. 104–112, Jun. 2019.
[16] P. K. Dash, A. K. Pradhan, G. Panda, and A. C. Liew, "Adaptive relay setting for flexible AC transmission systems (FACTS)," *IEEE Trans. Power Del.*, vol. 15, no. 1, pp. 38–43, Jan. 2000.
[17] B. Ravikumar, D. Thukaram, and H. P. Khincha, "An approach using support vector machines for distance relay coordination in transmission system," *IEEE Trans. Power Del.*, vol. 24, no. 1, pp. 79–88, Jan. 2009.
[18] A. N. Sheta, B. E. Sedhom, A. Pal, M. S. E. Moursi, and A. A. Eladl, "Machine-learning-based adaptive settings of directional overcurrent relays with double-inverse characteristics for stable operation of microgrids," *IEEE Trans. Ind. Informat.*, vol. 21, no. 2, pp. 1270–1282, Feb. 2025.
[19] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. Cambridge, MA, USA: MIT Press, 2009.
[20] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.
[21] N. Thomas, L. Hankins, and J. Perez, "Auto-tuned solution to wide area coordination issues of distance and directional time overcurrent relay settings," in *Proc. 74th Conf. Protective Relay Engineers (CPRE)*, College Station, TX, USA, 2021.
[22] J. Perez, A. Krislock, L. Hankins, and L. Lamb, "Distance and time overcurrent relay coordination with an autosolver framework," in *Proc. 74th Conf. Protective Relay Engineers (CPRE)*, College Station, TX, USA, 2023.
[23] "PRC-027-1: Coordination of protection systems for performance during faults," North American Electric Reliability Corporation (NERC), Nov. 2015.

VIII. Author Biographies

Reza Teshnizi is a computer scientist specializing in artificial intelligence, software design, and topological analysis. He earned his M.S. and Ph.D. in Computer Science from Texas A&M University in 2015 and 2024, respectively. He focuses on developing advanced computational methods and their real-world applications. Currently, he is the Vice President of Software at SynchroGrid, leading efforts in power systems software development.

Luke Hankins graduated from Texas A&M University with a B.S. degree in Electrical Engineering in 2015. He is a registered professional engineer in the state of Texas and works for SynchroGrid in relay setting automation. He writes software applications in various programming languages, focusing on simplifying relay setting development. Luke also performs relay settings verification and mis-operation analysis.