

ORIGINAL RESEARCH

Open access

Bridging AI Innovation and Regulatory Compliance in Financial Services: A Taxonomy of Regulatory Principles

Yifan Zhao^{1*}, Liang Chen¹, Ahmad Fauzi², Nor Hayati³

Abstract

The critical nature of explainable artificial intelligence and machine learning (XAI/XML) is gaining widespread recognition. This field focuses on uncovering how input variables influence automated choices, identifying systemic algorithmic bias, and detecting vulnerability to anomalies in data. Much of the existing literature centers on creating post hoc explanations tailored for opaque, black box architectures. These secondary techniques introduce novel vulnerabilities to error without correcting the fundamental flaws of the core model. Conversely, the development of inherently transparent systems receives far less attention. This paper introduces an inherently interpretable strategy rooted in the principles of ensemble learning, designated as the League of Experts (LoE) framework. We construct the initial theoretical foundations and subsequently derive a flexible, modular meta-algorithm. Although our presentation focuses primarily on classification tasks, the LoE framework adapts identically to regression scenarios. For classification specifically, we deploy traditional decision trees to form classifier ensembles as a concrete implementation. This paradigm simplifies the extraction of human-comprehensible decision rules for the underlying classification task, yielding an independent rule-induction system called RuleLoE. Beyond evaluating 12 KEEL classification datasets, we validate the LoE framework using two benchmarks from highly critical domains—healthcare and finance—to demonstrate its utility. The predictive capability of LoE matches contemporary state-of-the-art classification algorithms regarding both accuracy and rule coverage. Furthermore, LoE outputs a highly transparent group of decision rules featuring customizable complexity to map out the target classification problem. LoE provides a dependable approach for handling classification and regression problems. Its predictive precision is highly suited for environments where discovering root causal mechanisms is the primary objective, rather than merely maximizing predictive or classification performance.

Keywords Explainability, Ensemble learning, Multiagent systems, Glass box models

*Correspondence:

Yifan Zhao

yifan.zhao@cau.edu.cn

¹ Department of AI Innovation and Regulatory Compliance, Faculty of Engineering, China Agricultural University, Beijing, China

² Department of Financial AI and Regulatory Taxonomies, Faculty of Engineering, University of Malaya, Kuala Lumpur, Malaysia

³ Department of Regulatory Principles and AI Governance, Faculty of Engineering, Universiti Putra Malaysia, Serdang, Malaysia

Introduction

Machine learning (ML) has integrated into numerous daily applications and modern technologies, underlying progress in self-driving vehicles [1], medical diagnostics [2], and policing [3]. Legal frameworks regulating automated algorithmic determinations—such as “the right for explanation[s]” [4]—alongside other statutory considerations [5] highlight the vital need for broad transparency [6] in ML-centric systems, particularly regarding black box models. Black box systems maximize predictive metrics by compounding structural complexity to a degree that prevents human interpretation.

Current research endeavors attempt to counter this lack of transparency by layering on external systems that translate internal logic into human-readable data through added tiers of complexity (post hoc methods). However, these secondary layers function merely as rough approximations of the true decision-making pipeline and run the risk of being unfaithful to the core model [7]. Consequently, interest in explainable artificial intelligence and machine learning (XAI/XML) is expanding rapidly [8–11]. While various deployment sectors reflect the rising prominence of XAI and XML [12–14], the inherent trade-offs between accuracy and interpretability remain central to current scientific inquiry [15, 16].

Contribution of this work

This study presents the League of Experts (LoE), an original and transparent machine learning model and meta-algorithm built upon the concepts of ensemble learning [17] (p. 605). LoE operates as a specialized variant of dynamic classifier selection (DCS) frameworks [18]. Crucially, whereas conventional DCS implementations typically generate a redundant pool of classifiers using static routines like bagging or boosting [19], LoE evolves its ensemble directly during the selection and assignment phases. To our knowledge, this paradigm has not yet been investigated within the DCS literature [18].

LoE serves as an architectural blueprint that facilitates modifications tailored to maximize comprehensibility. It allows developers to create user interfaces that let human operators interact directly with the model and its subcomponents. Although this paper does not delve into specific interface configurations, it highlights potential avenues for interaction where LoE permits humans to manually tune both the selection routines and the individual ensemble units. In this manner, LoE enables users to interactively investigate the balance between model accuracy and cognitive complexity.

In this paper, we demonstrate and benchmark LoE using two distinct datasets. We focus specifically on regulating the degree of structural complexity needed to clarify the learned model instances. In addition, we present a concrete architectural configuration to implement LoE, enabling its conversion into a virtually equivalent rule set learner. To minimize structural complexity while simultaneously sharpening the approximation of the base LoE model, we further investigate methods to modify the training protocol.

Materials and Methods

In this section, we formulate the League of Experts (LoE) algorithm, detailing its design motivations regarding transparency and accuracy. To anchor this discussion, we outline the background context and taxonomy of historical and modern strategies explored within XAI and XML. T

Black box models

Computing innovations have driven the use of increasingly powerful but structurally convoluted models known as black box models [20–24]. For example, feed-forward artificial neural networks (ANNs) applied to image classification tasks often contain roughly 5 to 155 $\times 10^6$ optimizeable parameters while performing up to 8 $\times 10^{10}$ individual mathematical calculations for a single forward pass [25]. Because an ANN is structured as a directed graph of operations, it is technically feasible to trace every computation sequentially. In practice, however, such vast structural complexity cannot be meaningfully processed by human cognition.

Even alternative popular models that utilize significantly fewer parameters, such as support vector machines (SVMs) [26] or ensemble architectures like boosting (i.e., AdaBoost [27]) and random forests [28], defy simple interpretation. These systems are poorly suited for applications that demand a clear understanding of the decision-making logic. In ensemble learning, this limitation holds true even if the individual ensemble components are inherently interpretable, such as decision trees [29]. This problem persists because the decision boundary of the collective ensemble remains highly intricate, since the final output is generated by aggregating predictions across all participating members.

Glass box models

By contrast, glass box models [22, 30, 31], alternatively known as white box models, feature transparent internal operations that allow users to derive explicit explanations regarding their decision boundaries. This category includes decision trees [17] (p. 305), (logistic) regression [17] (p. 119), and instance-based neighborhood algorithms that explain predictions through the proximity of related data points, such as k-nearest neighbors (k-NN) [32].

However, introducing excessive complexity compromises practical intelligibility—such as deep decision trees with too many levels or Bayesian networks featuring excessive dependencies [33]. Crucially, this limitation stems from the cognitive boundaries of human users rather than from the mathematical mechanics of the decision-making process itself.

Rule learning

An illustrative instance of glass box structures is decision rules, which present analytical insight to users via intuitive logical configurations. These configurations are developed either as unordered (i.e., independent) structures known as rule sets, or as ordered sequences designated as rule lists. The latter are considerably more difficult to interpret [34] because the validity of any single rule relies on the negation of all preceding entries (meaning prior rules must fail to apply).

Decision trees can be converted into rule sets by translating each individual path from the root to a leaf into a distinct rule. For instance, the logical structure illustrated in Figure 1 corresponds to the following formulation: if $(\text{sex is female} \wedge \text{age} > 40 \wedge \text{weight} > 80) \vee (\text{sex is female} \wedge \text{age} \leq 40)$, predict low-risk group; otherwise, predict high-risk group. In contrast to decision trees, most rule induction frameworks do not generate rule sets characterized by exhaustive coverage and mutually exclusive sample space partitioning—meaning that incoming queries may satisfy either multiple rules or no rules at all. The dilemma of multiple matching rules is resolved by establishing a definitive execution order, whereas the absence of matching rules is handled by introducing a catch-all fallback rule. Both design challenges must be resolved when adapting the LoE framework into an independent rule learner

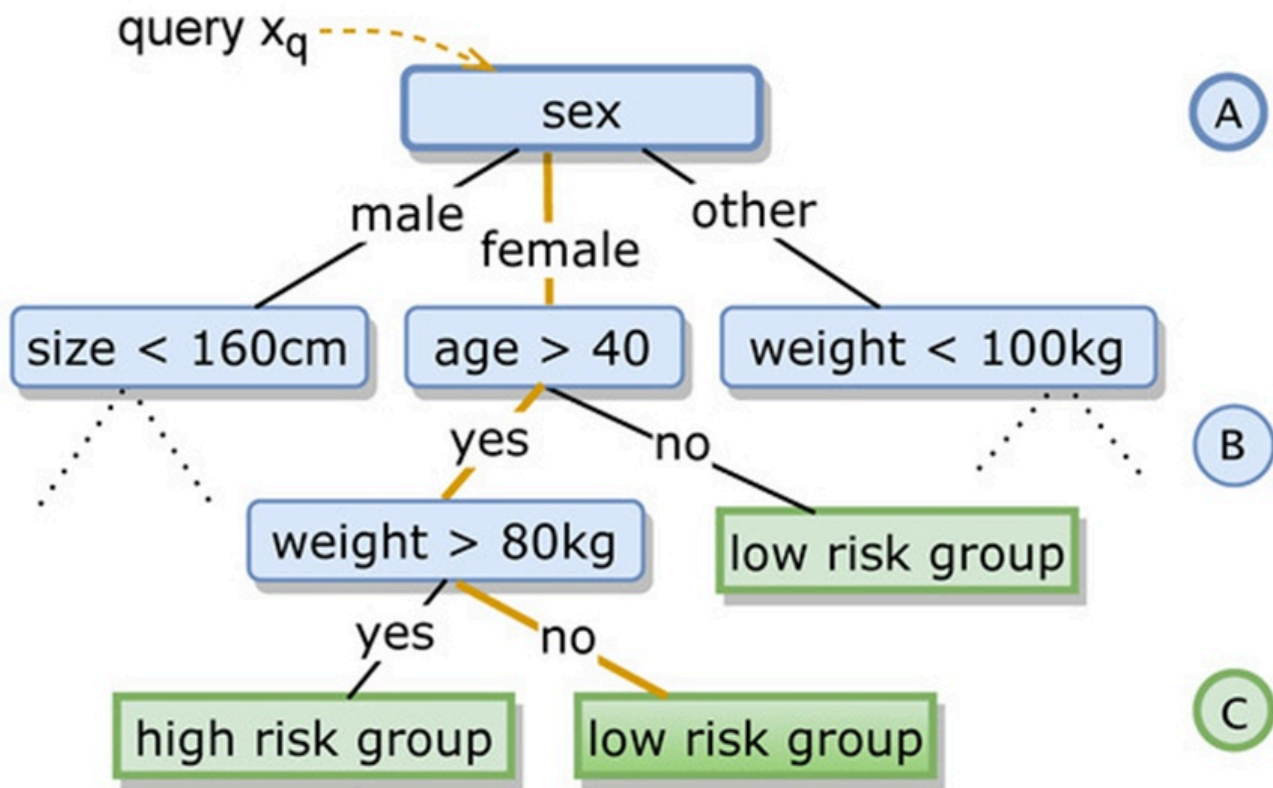


Figure 1. (Partial) decision tree example with (A–C) being the root, inner nodes, and leaves, respectively. An example query with corresponding trace is highlighted in orange.

Rule learning architectures, such as CN2 [35], RIPPER [36], Bayesian rule sets [37], Boolean decision rules via column generation (BRCG) [38], and interpretable decision sets [34], extract logical statements by directly maximizing an objective function. This optimization frequently utilizes integer programming to satisfy targeted metrics such as classification performance, coverage, or structural simplicity. Identifying the mathematically optimal solution is often computationally intractable due to the exponential growth of potential logical clauses, which reduces the task to the NP-complete set cover problem.

Explanations

Within the XAI domain, the term “explanation” signifies human-comprehensible data evaluated from a local or global perspective, serving to clarify either (i) the underlying mechanism that produces an output for a specific query (such as a classification assignment) or (ii) the operational state of the entire model architecture. Local explanations can yield deceptive insights regarding the global behavior of a system, given that query-specific characteristics often do not generalize to the broader global context, i.e., across arbitrary inputs.

Post hoc explainers represent a widely used paradigm. These frameworks introduce supplementary diagnostic tools and visual interfaces designed to clarify the decision logic of a previously optimized model. Post hoc strategies are classified as either (i) model-agnostic (fully independent of the underlying algorithm class), such as LIME [39], Anchors [40], Shapley values [41], or graphical techniques like partial dependence plots (PDPs) and individual conditional expectation plots (ICE) [42]; or (ii) model-specific, which include decision-tree-tailored SHAP modifications (TreeSHAP) [43] or specialized neural network interpretability tools [44].

However, post hoc strategies exhibit notable limitations: (i) the integration of external diagnostic layers compounds the complexity of machine learning pipelines, and (ii) users must place trust in both the accuracy of the underlying machine

learning model and the validity of the explanation framework itself. Either component can be erroneous, flawed, or valid only within specific unrecorded bounds. Crucially, human operators cannot easily detect these failures. Transparent glass box models bypass the need for post hoc interventions entirely, a characteristic shared more broadly by globally simulatable models that are sufficiently compact to be comprehended by a user within a reasonable timeframe [45].

Simulatable models encompass compact decision trees, restricted rule sets [46], (generalized) linear models containing a limited number of coefficients (achieved through regularization methods, e.g., Lasso), or instance-based estimators utilizing easily interpretable features (e.g., k-NN [32]). Furthermore, the features characterizing the underlying data instances must be intrinsically understandable. Models that conform to these rigorous constraints are highly restricted, often exhibiting lower predictive performance and constrained generalization capabilities [47]. Notably, the precise definition of what constitutes a simulatable model remains somewhat subjective, as it is bounded by the cognitive capacity of individual users.

Decision trees

Decision trees [48] directly reveal their internal logic via an intuitive graphical architecture (Figure 1). They typically isolate and evaluate only a small subset of highly informative variables or features [49], which minimizes cognitive demand on the observer. This graphical framework allows human users to closely scrutinize the nuances of a specific inference path, while also enabling the exploration of counterfactual scenarios—i.e., modifying an attribute value to observe how the downstream decision pathway alters. This capacity allows for rigorous auditing of both the fairness and the contextual relevance of the model's outputs.

Surrogate models

Surrogate (or proxy) frameworks [50] seek to illuminate the inner workings of a complex black box model $f(\cdot)$ by replicating its predictive output profile across a dataset $\mathcal{D}$ using a simpler, inherently interpretable glass box model $f_g(\cdot)$ [42] (for instance, employing a shallow decision tree to mimic a deep ANN). This mapping function $f_c \sim f_g$ can be designed to hold globally (a global surrogate) across the entire dataset $\mathcal{L}$, or locally (a local surrogate) within a specific subregion of $\mathcal{L}$. As an example, the LIME technique [40] functions as a local surrogate model.

In general, it remains difficult to verify whether the proxy model f_g captures the exact causal relationships leveraged by the primary system f_c . A variety of structurally distinct surrogates might achieve identical approximation accuracy, a phenomenon explained by the Rashomon effect [51], which describes scenarios where multiple conflicting explanations yield identical predictive outcomes. This issue is less problematic when substituting the complex architecture with a highly precise global surrogate. For local surrogates, however, these discrepancies compound significantly, particularly if users inadvertently extrapolate global decision logic from localized explanations. While these structural vulnerabilities cannot be completely eliminated, they can be controlled and minimized by strategically guiding the training workflow.

Ensemble learning and dynamic selection

Ensemble learning strategies [52, 53] optimize a collection of multiple base models to solve classification or regression objectives, combining their outputs via distinct aggregation or fusion mechanisms. The LoE framework aligns directly with this operational paradigm. More specifically, it represents a dynamic selection (DS) methodology [18], which diverges from static approaches (such as AdaBoost [27] and random forests [28]) by refraining from querying the entire pool of models to construct a final prediction. Instead, it utilizes an independent selection routine to isolate specific ensemble subsets for each individual input query.

The core philosophy of DS is to dynamically identify a specialized subset of models that possess high localized expertise for a given query submitted to the system. This is achieved by utilizing a filter routine that evaluates which ensemble constituents are best suited to predict a given input. This filtering routine is frequently designated as a selection or assignment function. DS methodologies are broadly classified as either dynamic classifier selection (DCS) or dynamic

ensemble selection (DES) frameworks. While DCS techniques isolate exactly one model from the available pool when evaluating an incoming query, DES approaches can select a multi-model sub-ensemble. LoE is formally classified as a DCS method, as it isolates exactly one base model per query to maximize structural explainability.

The League of Experts Classifier

To implement the League of Experts as a classification system, we modify a specific instance of ensemble learning where individual ensemble components are mapped to designated regions of the feature space. More precisely, we present an algorithmic approach that dynamically constructs a partition of the feature space to allocate a distinct, specialized ensemble component—termed the expert—to each respective subregion of the partition. Because this partitioning segments the feature space into mutually exclusive subsets, each expert undergoes training on a distinct portion of the training dataset. This process introduces diversity into the learning mechanism, implying that individual experts achieve high performance specifically within their allocated regions, but not necessarily across the entirety of the feature space. Given that individual experts do not need to capture the global characteristics of the feature space, their internal complexity can be restricted without compromising overall system performance.

The terminology for this ensemble of experts is derived from the manner in which it executes inference. Following the training of an LoE classifier, the inference phase operates via a two-stage procedure: (i) a given query x_q is directed to the appropriate expert g , and (ii) the resulting query prediction $g(x_q)$ is computed by the designated expert (as illustrated in Figure 2). This framework mirrors cooperative human problem-solving, where multiple specialists operate within distinct domains, and a supervisor delegates assignments to the most qualified authority. As an analogy, consider a query as a patient visiting a general practitioner (acting as the coordinator) for an initial assessment. In this scenario, the coordinator routes the patient to the specialized medical professional.

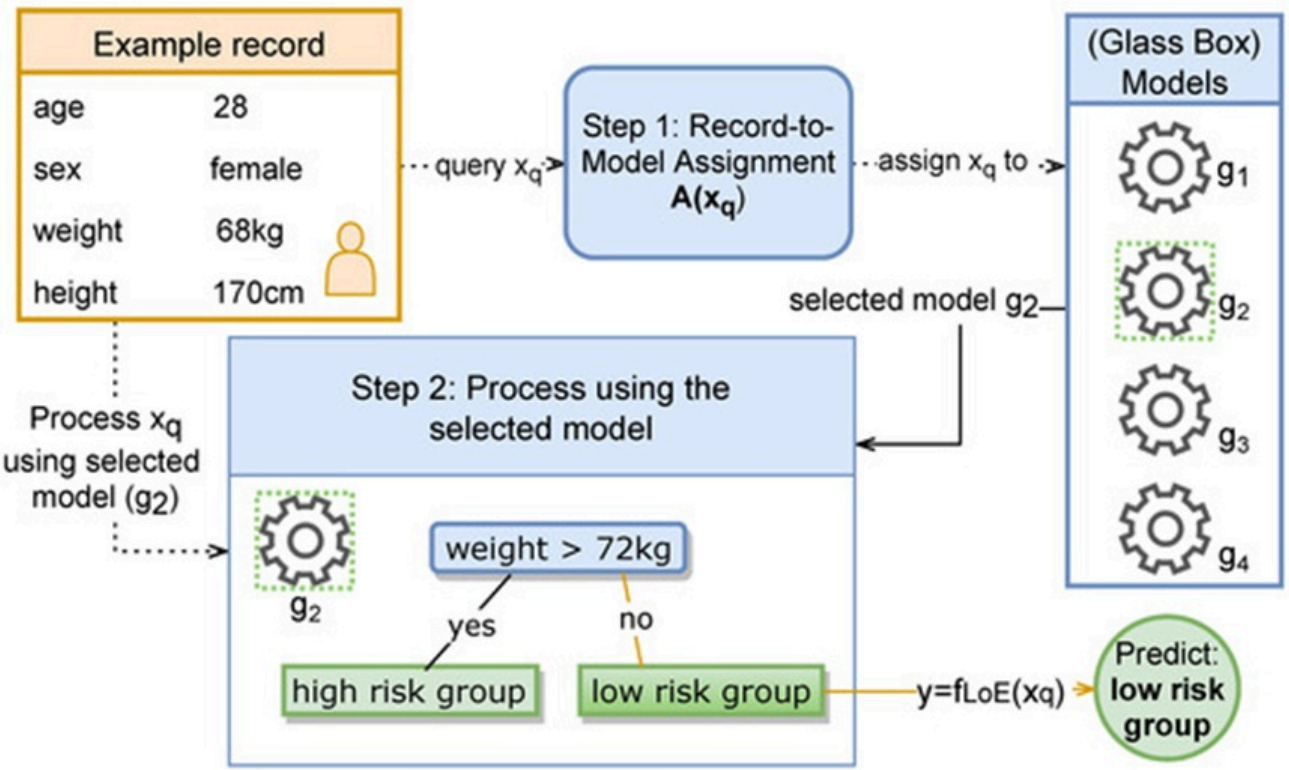


Figure 2. LoE’s two-step decision process: (i) expert assignment to query (here, g_2 , a one-level decision tree) and (ii) query prediction by the assigned expert (here, prediction low-risk group). Here, experts are illustrated as gear wheels.

The subsequent subsections detail our formalizations and model architecture, our framework for generating interpretability, the minimization of explanatory complexity, the evaluation of LoE's computational runtime complexity, and the rule-based learning variant of LoE, denoted as RuleLoE .

Basic definitions and model formulation

Let us examine a standard supervised learning scenario centered on a labeled dataset $\mathcal{L} \subseteq \mathcal{X} \times \mathcal{Y}$. Within this framework, each labeled instance $(x, y) \in \mathcal{L}$ is drawn from a probability distribution and comprises m real-valued features within a domain $\mathcal{X} \subseteq \mathbb{R}^m$, alongside a categorical (or, for regression tasks, continuous) target label belonging to a set $\mathcal{Y}$ (where continuous targets satisfy $\mathcal{Y} \subseteq \mathbb{R}^1$). It should be noted that categorical features are permissible, provided they undergo one-hot encoding. For any given instance $(x, y) \in \mathcal{L}$, the objective is to maximize the conditional probability of the target variable Y given the feature vector X utilizing the model $P(Y=y|X=x)=g_s(x, y)$, where $g_s: \mathcal{X} \times \mathcal{Y} \rightarrow [0, 1]$ represents a probabilistic (soft) classifier chosen from a family $\mathcal{G}_s$.

The corresponding deterministic (hard) classifier derived from this formulation is expressed as:

$$g(x) := \arg \max_{y \in \mathcal{Y}} P(Y = y | X = x), \quad (1)$$

where $g: \mathcal{X} \rightarrow \mathcal{Y}$. Let $\mathcal{G}$ denote the family of hard classifiers derived from $\mathcal{G}_s$. When addressing regression tasks instead of classification, the model targets the conditional expectation of the output Y given the input X rather than the conditional probability. Specifically, the underlying regression formulation is expressed as $(Y|X=x)=g(x)$.

The mapping defined by the following function:

$$A: \mathcal{X} \rightarrow \{1, \dots, n\}$$

is designated as an assignment function of order n ($|A|=n$). Every assignment function A establishes a partition $\mathcal{X} = \bigcup_{k=1}^n \mathcal{X}_k^{(A)}$ such that $\mathcal{X}_k^{(A)} \cap \mathcal{X}_l^{(A)} = \emptyset$ for $k \neq l$ across the feature space $\mathcal{X}$ of rank n via the following relation:

$$\mathcal{X}_k^{(A)} := A^{-1}(k). \quad (2)$$

By allocating a single classifier (g_k) to each distinct subregion ($\mathcal{X}_k^{(A)}$) of the partition, we generate an n -tuple $\mathcal{G}^*$ in $\mathcal{G}^n$ (derived from $\mathcal{G}_s^* \in \mathcal{G}_s^n$). The ordered pair $(\mathcal{G}^*, A)$ constitutes an LoE of order n determined by A . The subregion $\mathcal{X}_k^{(A)}$ is referred to as the territory of the corresponding expert g_k .

An LoE establishes a singular composite classifier via the mapping:

$$x \mapsto g_{A(x)}(x),$$

which typically does not belong to the original family $\mathcal{G}$. The space of all valid assignment functions $\mathcal{A}$ (or equivalently, the space of all corresponding geometric partitions $\mathcal{A}_{\text{part}}$) along with the classifiers $\mathcal{G}$ defines the domain of all potential LoE classifiers, represented by $\mathbb{G}$. It is worth noting here that the LoE framework shares conceptual similarities with DCS algorithms, where the assignment function serves the purpose of the selection mechanism. Moving forward, because any assignment function uniquely corresponds to the partition it constructs, we treat assignment functions and their corresponding partitions interchangeably.

The accuracy of an LoE classifier is evaluated using a utility metric $p: \mathcal{C} \times \mathcal{C} \rightarrow \mathbb{R}$. A standard implementation is the Kronecker delta function:

$$p(\hat{y}, y) := \begin{cases} 1 & \text{if } \hat{y} = y, \\ 0 & \text{if } \hat{y} \neq y. \end{cases} \quad (3)$$

For continuous targets (regression operations), the utility metric is generally derived from a distance calculation between $\hat{y}$ and y , such as the negative mean-squared error.

An optimal LoE classifier maximizes the expected utility across the joint input and output space $\mathcal{X} \times \mathcal{C}$, that is, it satisfies the following optimization problem:

$$f_{\text{LoE}}(x) := \arg \max_{f \in \mathcal{G}} \int_{\mathcal{X} \times \mathcal{C}} p(f(x), y) dF(x, y), \quad (4)$$

where $F(x, y)$ represents the joint probability distribution governing inputs X and outputs Y .

The maximization problem defined in (2) is resolved via a two-tiered process: (i) for every individual assignment function $A \in \mathcal{A}$ (or its corresponding partition), an optimal expert collective $\mathcal{G}_A^* = \left(g_k^{(A)} \right)_{k=1}^{|A|}$ is determined, and (ii) these localized optimal configurations $(\mathcal{G}_A^*)$ are subsequently maximized across the entire space of assignment functions $A \in \mathcal{A}$ (or all valid partitions). This procedure yields the globally optimal LoE classifier $(\mathcal{G}_{A'}^*, A')$, expressed as:

$$f_{\text{LoE}}(x) := g_{A'(x)}^{(A')}(x), \quad (5)$$

where

$$A' = \arg \max_{A \in \mathcal{A}} \int_{\mathcal{X} \times \mathcal{C}} p(g_{A(x)}^{(A)}(x), y) dF(x, y), \quad (6)$$

and

$$\mathcal{G}_A^* = \left(g_k^{(A)} \right)_{k=1}^{|A|} = \arg \max_{\mathcal{G}^* \in \mathcal{G}^{|A|}} \int_{\mathcal{X} \times \mathcal{C}} p(g_{A(x)}(x), y) dF(x, y). \quad (7)$$

It is important to emphasize that an optimal LoE classifier is not guaranteed to be unique. Furthermore, computing such a classifier exactly is practically impossible, because the true underlying distribution $F(x, y)$ is unknown and the space of potential partitions is combinatorially exhaustive. In real-world applications, an empirically pseudo-optimal LoE is resolved by substituting $F(x, y)$ with the empirical distribution function $\hat{F}(x, y)$ derived from the observed training dataset $\mathcal{L}$. This yields the empirical optimization formulation:

$$f_{\text{LoE}}(x) := g_{A'(x)}^{(A')}(x), \quad (8a)$$

where

$$A'(x) = \arg \max_{A \in \mathcal{A}} \frac{1}{|\mathcal{L}|} \sum_{(x,y) \in \mathcal{L}} p(g_{A(x)}^{(A)}(x), y), \quad (8b)$$

and for each $A \in \mathcal{A}$

$$\mathcal{G}_A^* = \arg \max_{\mathcal{G}^* \in \mathcal{G}^{|\mathcal{A}|}} \frac{1}{|\mathcal{L}|} \sum_{(x,y) \in \mathcal{L}} p(g_{A(x)}(x), y). \quad (8c)$$

Nonetheless, solving this empirical optimization remains computationally intractable. Generally, the spaces containing the tuples $\mathcal{G}^* \in \mathcal{G}^{|\mathcal{A}|}$ (6b) and the assignment functions $A \in \mathcal{A}$ (6a) are uncountably infinite. However, a heuristic approximation can be achieved for restricted families of assignment functions. We introduce an algorithmic framework that produces an estimated LoE $(\widehat{G}^{(A^*)}, A^*)$ whose performance closely approximates that of the true optimal model.

Class of assignment functions

The collection of assignment functions $\mathcal{A}$ of order n is confined to the specific class of functions parameterized by n pairwise distinct anchor points $\theta_1, \dots, \theta_n \in \mathbb{R}^m$ according to the following relation:

$$A : \mathcal{X} \rightarrow \{1, \dots, n\}; \quad A(x) := \min_{k \in \{1, \dots, n\}} \left(\arg \min d(x, \theta_k) \right),$$

where $d(\cdot, \cdot)$ represents a specified metric over $\mathcal{X}$. Without incorporating the minimization operation, A would be ill-defined for instances where $(x, \cdot) = d(x, \theta_i)$ given $k \neq i$. Consequently, we establish the following condition:

$$\begin{aligned} \mathcal{A}_n := \{ A \mid A(x) := \min_{k \in \{1, \dots, n\}} \left(\arg \min d(x, \theta_k) \right) \\ \text{for pairwise different } \theta_k \in \mathbb{R}^m, k = 1, \dots, n \}. \end{aligned} \quad (9)$$

Within this framework, $\mathcal{A}_n$ is contingent upon the selected metric d . It is recommended to choose a metric that accounts for the scale and variability of the input attributes. In this study, we utilize the weighted L_1 metric $(x, \cdot) = \sum_{i=1}^m |x_i - y_i| / \sigma_i$, where σ_i denotes the empirical standard deviation corresponding to the i -th feature. In practical applications, σ_i is substituted by its corresponding estimate computed from the labeled dataset $\mathcal{L}$. An alternative approach would involve utilizing the Mahalanobis distance.

Because a bijective relationship exists between the assignment function A and an n -tuple of anchor points $(\theta_1, \dots, \theta_n)$ for every $A \in \mathcal{A}_n$, these two representations are treated interchangeably. Within an LoE classifier architecture, anchor points denote the central locations of the domains governed by each expert, which is why they are designated as such.

Algorithm training

An assignment function A segments not only the continuous feature space $\mathcal{X}$ but also the discrete labeled dataset $\mathcal{L}$. Specifically,

$$\mathcal{L}_k^{(A)} := \{(x, y) \in \mathcal{L} \mid x \in \mathcal{X}_k^{(A)}\}. \quad (10)$$

We approximate the optimization problem formulated in (6) for the class of assignment functions $\mathcal{A}_n$ by utilizing the heuristic optimization routine diagrammed in **Figure 3**. This procedure initializes with n starting anchor points $\theta_1, \dots, \theta_n$ and their corresponding assignment function A . Following this, equation (6b) is approximated with respect to the current assignment function A by optimization of a single classifier $g_k \in \mathcal{G}$ using the corresponding subset of the partitioned labeled dataset $\mathcal{L}$. This training yields an ensemble $\widehat{\mathcal{G}}_A^*$, thereby establishing an LoE configuration $(\widehat{\mathcal{G}}_A^*, A)$ that is "quasi-optimal for A ". Next, the anchor points θ_k undergo an adjustment phase. We translate the central coordinates of the experts' domains, substituting θ_k via the following update rule:

$$\theta'_k = \theta_k + \frac{\eta}{|\mathcal{L}|} \sum_{(x,y) \in \mathcal{L}} (x - \theta_k) p(g_k(x), y), \quad (11)$$

where η represents the operational step size of the update. During this phase, the performance of each individual expert is quantified across the entirety of the training dataset $\mathcal{L}$. The domain centers θ_k are systematically shifted toward regions where the specific expert g_k demonstrates high accuracy as quantified by the performance metric $((\cdot), \cdot)$. These adjusted anchor locations establish a modified assignment function A' along with a new partition of the underlying feature space, for which a subsequent LoE configuration is trained. This iterative step is repeated, often using systematically decaying step sizes. Ultimately, the specific LoE model that achieves the highest global performance metric is selected, which is designated as the "quasi-optimal LoE". The programmatic logic of this approach is formalized as pseudocode in **Algorithm 1**. The operational mechanics of the adaptation step are fundamentally dependent on the chosen performance metric. For the sake of simplicity, the Kronecker delta function (1) is implemented for the empirical evaluations detailed herein.

Input:

$\mathcal{G}$: set/class of (explainable) models, e.g., decision trees;
 $\mathcal{L}$: Labeled dataset with $(x, y) \in \mathcal{L}, x \in \mathcal{X}, y \in \mathcal{C}$;
 n : degree of LoE classifier;
 η : step size;
 T : number of iterations;
 d : metric/distance, e.g., $d(x, x') = \|x - x'\|_2$;
 p : performance function, e.g., $p(y, y') = \delta(y, y')$.

Result:

$(\hat{\mathcal{G}}^{(A^*)}, A^*)$: trained quasi-optimal LoE classifier, i.e.,
 expert ensemble $\mathcal{G}^* = \{g_1^*, \dots, g_n^*\}$, territories'
 center points $\theta_1^*, \dots, \theta_n^*$, defining A^*

Initialize:

$S \leftarrow \emptyset$ # stores pairs of experts and center points
 $S' \leftarrow S, i \leftarrow 0, p' \leftarrow -\infty$
 # initialize experts at random position in $\mathcal{X}$
 $S = \bigcup_{i=1}^n \{(g_i, \theta_i)\}$

while $t < T$ **do**

 # train experts on their territories:
 foreach $(g_i, \theta_i) \in S$ **do**
 $\mathcal{L}_g \leftarrow \{(x, y) | d(\theta_i, x) \leq d(\theta_j, x), j = 1, \dots, n\}$
 Fit model g_i on dataset $\mathcal{L}_g$
 end
 foreach $(g_i, \theta_i) \in S$ **do**
 # move territories' center points
 $\theta'_i \leftarrow 0|_1^m$ # stores the movement vector
 foreach $(x, y) \in \mathcal{L}$ **do**
 # move towards x weighted by performance
 $\theta'_i \leftarrow \theta'_i + p(g_i(x), y) * (x - \theta_i)$
 end
 $\theta_i \leftarrow \theta_i + \eta / |\mathcal{L}| * \theta'_i$
 end
 $P' \leftarrow 0$ # calculate current performance
 foreach $(x, y) \in \mathcal{L}$ **do**
 # inference using the current experts and positions
 $y' \leftarrow f_{\text{LoE}}(x)$
 $P' \leftarrow P' + p(y, y')$
 end
 # keep track of best performing LoE
 if $P' > P$ **then**
 $P \leftarrow P'$
 $S \leftarrow \bigcup_{i=1}^n \{(g_i, \theta'_i)\}$
 end
 $t \leftarrow t + 1$

end

Return: S

Algorithm 1. Basic League of Experts Training Procedure

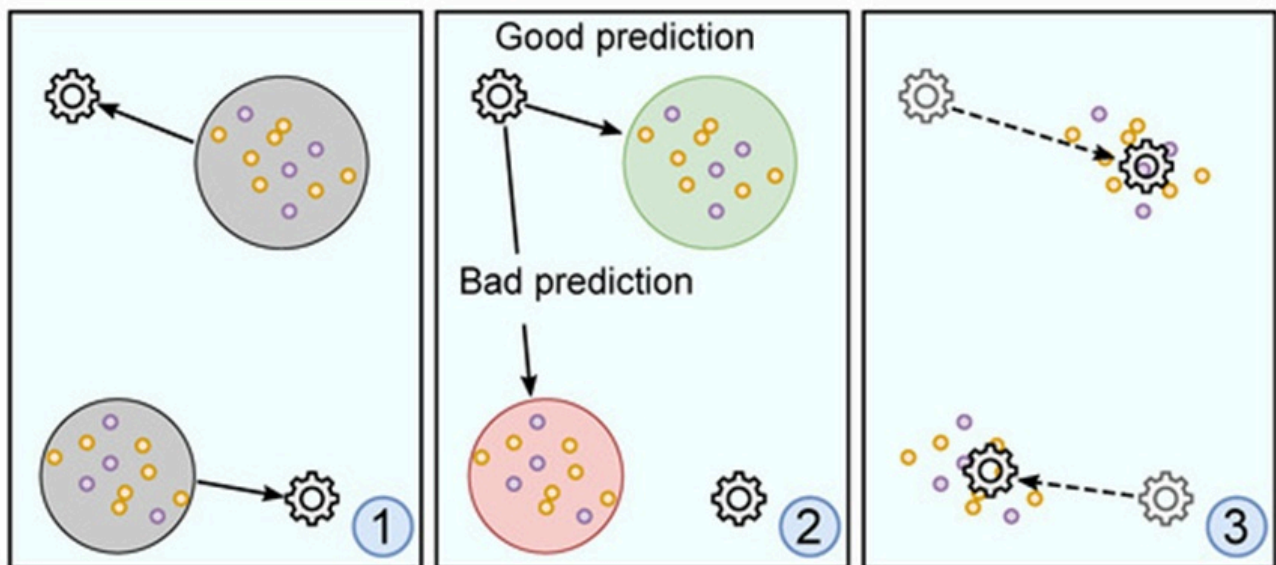


Figure 3. Training process of LoE with two experts separated into three steps: (1) data-to-model assignment (A), (2) evaluation phase, and (3) optimization phase (movement). Experts are illustrated as gear wheels. The training data are shown using purple and yellow circles, whereas each color depicts a distinct class of the learning problem. In (2), a model's expected prediction performance on different partitions of the feature space are colored in green and red, relating to good and a bad performance, respectively.

LoE inference

Let $(\hat{\mathcal{G}}(A^*), A^*)$ represent a quasi-optimal LoE framework where A^* is defined by the final anchor positions $\theta_1^*, \dots, \theta_n^*$. The resulting tuple $(\hat{\mathcal{G}}(A^*), A^*)$ serves as an operational replacement for f_{LoE} in Eq. 3. For any incoming query x_q , the respective distances (x_q, θ_k^*) are evaluated for all $k=1, \dots, n$. Subsequently, x_q is routed to the corresponding closest specialized component, denoted as g_i^* in Eq. 7, to calculate the final target prediction $\hat{y} = g_i^*(x_q)$ (Figure 2).

Deriving explanations

In the preceding sections, we outlined a highly modular and functional machine learning architecture. When configured using glass-box models as the base experts, the framework itself is capable of optimizing an array of distinct (by virtue of being trained on mutually exclusive data partitions) and collaborative classification models (cooperating to resolve a shared target task). Although the optimized experts can independently uncover insights regarding the underlying learning task by explicitly modeling a localized component of the decision boundary, the global logic governing the assignment function remains opaque. The subsequent sections address this limitation by exploiting the structural characteristics of the architecture to extract comprehensive explanations.

Within our framework, interpretability encompasses two distinct dimensions: (i) clarifying the assignment mechanics, which explains why a given query is routed to a particular expert, and (ii) interpreting the internal logic of the experts themselves, which explains why an expert generates a particular prediction within its allocated domain. The latter aspect necessitates that the base experts are structurally transparent glass-box models—the paradigm adopted in this study—or alternatively, that post-hoc explainability frameworks are integrated. To clarify the assignment mechanics, the underlying decision boundaries generated by the assignment function A must be interpretable, a task that can be highly non-trivial.

The resulting explanations must satisfy criteria of fidelity, interpretability, and exhaustiveness; these demands inherently create a trade-off, as expressed by the notion that "One Explanation Does Not Fit All [Alike]" [54].

While a wide variety of strategies exist to interpret an assignment function A (or its implicit geometric partitions), our primary objective centers on uncovering functional relationships and interactions among key features. To accomplish this, we convert the internal assignment logic into an equivalent auxiliary classification task. This surrogate task is then solved by training a glass-box model tasked with approximating the behavior of A . Although utilizing surrogate models to approximate an underlying pipeline introduces inherent fidelity risks, the LoE architecture possesses structural qualities that substantially mitigate these vulnerabilities. Most notably, a core segment of the model's predictive sequence remains completely uncompromised, because the localized experts themselves are never approximated by the surrogate system. By regulating the explicit complexity of these surrogate architectures, implementing varied modeling approaches, and integrating dimensionality reduction methods driven by a refined training pipeline, the fidelity of the surrogate approximation can be further optimized. Although a wide variety of predictive models can be adapted for this task, this investigation focuses primarily on the deployment of decision trees, as they facilitate a straightforward transformation of the trained LoE model into a structured rule-based system designated as RuleLoE.

Rendering the assignment function explainable

To provide transparency for the assignment function, an interpretable glass box model is developed to serve as a surrogate for the original assignment function A . Let $\mathcal{P}$ represent the family of these surrogate models, which necessitate a training dataset. The inputs fed into the assignment function consist of elements from the feature space $\mathcal{X}$, whereas the corresponding labels belong to the set $\{1, \dots, |A(\mathcal{D})|\}$. For a specific subset $\mathcal{X}_s \subseteq \mathcal{X}$ within the feature space, the graph representing the assignment function A restricted to $\mathcal{X}_s$ is defined as follows:

$$\tilde{\mathcal{L}}_{A, \mathcal{X}_s} = \{(x, A(x)) | x \in \mathcal{X}_s\}. \quad (12)$$

Surrogates function as assignment models themselves, though their resulting partitions generally do not belong to $\mathcal{A}$. A dataset of this $\tilde{\mathcal{L}}_{A, \mathcal{X}_s}$ form can be utilized as labeled data to train a specific surrogate $h \in \mathcal{P}$. This collection is referred to as the assignment dataset. Throughout the following sections, A represents the assignment function of a designated LoE $(\mathcal{G}^*, A)$. We structure A as a decision tree trained on the assignment dataset $\tilde{\mathcal{L}}_{A, \mathcal{X}_s}$ (meaning A is trained to map a given data point to a specific expert). The complete decision-making path for classifying a query instance x_q is formulated as a concatenation of the individual decision paths from the two models A and g_k , where $k=(x_q)$, while the final prediction is expressed as $\hat{y}=g_{A(x_q)}(x_q)$. This procedure is illustrated in **Figure 4**.

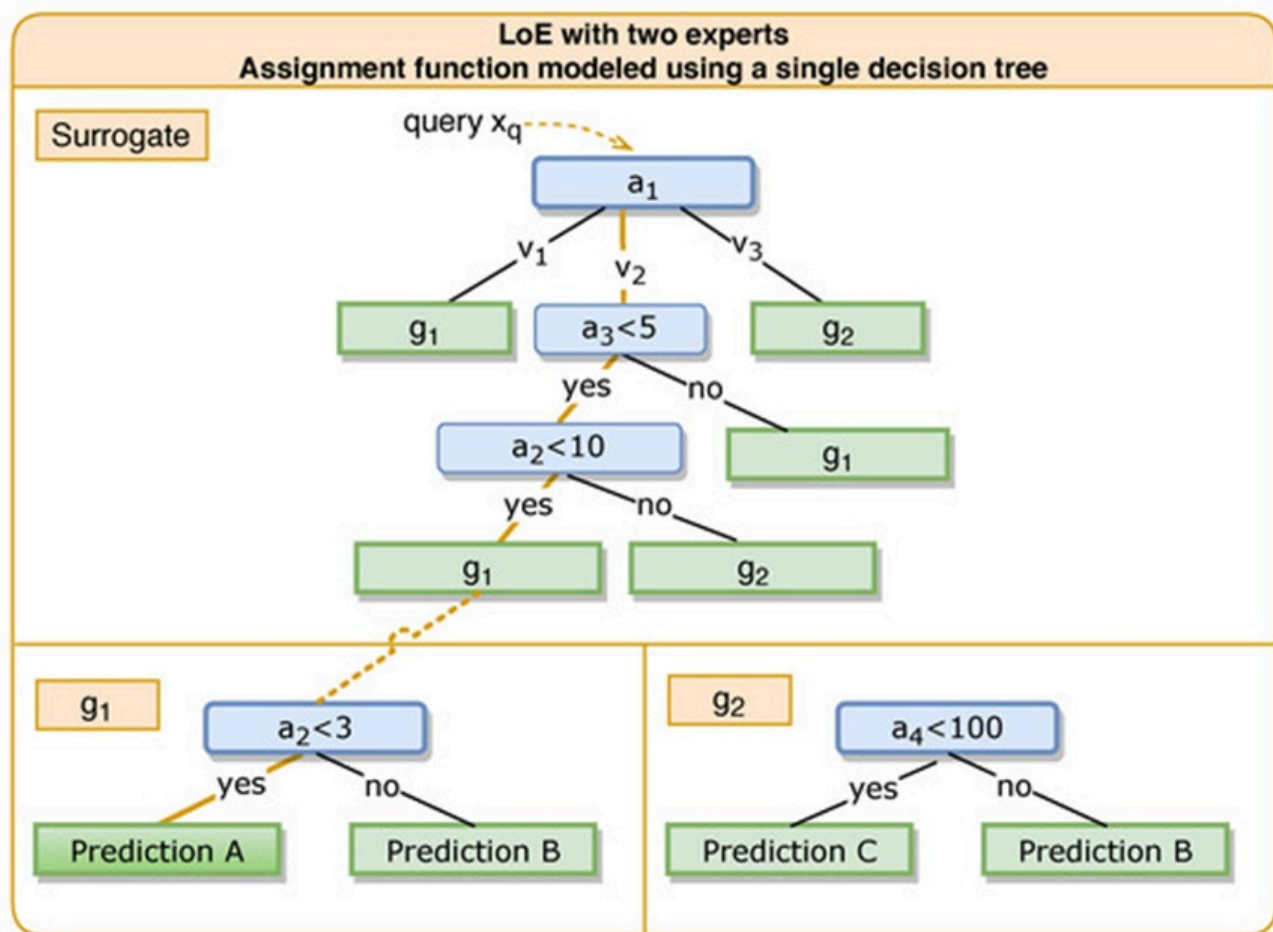


Figure 4. Exemplified LoE instance containing two experts (g_1 and g_2) with A being replaced by a single decision tree (surrogate). The orange line denotes a decision path of a query x_q (where $x_q = (a_1, a_2, a_3)$ satisfies $a_1 = v_2 \wedge a_3 < 5 \wedge a_2 < 3$) through both the surrogate and one expert. At this point, it is also noted that only the expert g_1 is relevant for the query x_q .

The surrogate's faithfulness

The capability of the surrogate to accurately replicate the baseline assignment function A (i.e., its assignment fidelity) must be assessed across a sample evaluation set $\mathcal{X}_s \subseteq \mathcal{X}$. The degree of consistency between the surrogate h and A is quantified via $p_{ass}(h, \mathcal{X}_s)$, where the metric function is defined as $(p_{ass}: \mathcal{H} \times \mathcal{A} \times (\mathcal{A}(\mathcal{X}) \setminus \emptyset) \rightarrow \mathbb{R}^{\geq 0})$ and $\mathcal{P}$ represents the power set. Higher values signify superior alignment.

If the surrogates display inadequate consistency, it implies that the model class $\mathcal{S}$ might fail to capture the same underlying patterns as the components in $\mathcal{A}$, or it encompasses the correct mechanisms but is overly restricted. In such scenarios, the class $\mathcal{S}$ can be broadened (e.g., by loosening regularization parameters, such as permitting greater tree depth). Alternatively, the dimensionality of the input features evaluated by the surrogate can be scaled down, for example, by narrowing their feature space. Relevant projections can be determined during the LoE optimization routine.

Because the primary objective is to generate intelligible explanations for the entire operational pipeline, the clarity of the surrogate must be maintained. Consequently, surrogates must remain comprehensible. Given that the selected decision trees can grow to arbitrary levels of complexity, a metric must be established to quantify this attribute. To achieve this, we first outline the fundamental notation.

Decision-tree-related notation

For a given decision tree t , its root node, the collection of terminating leaf nodes, and the ancestral path from a specific node v back to the root are designated as $\text{head}(t)$, $\text{leafs}(t)$, and $\text{trace}(t, v)$, respectively. The decision tree routes every individual data point to a specific leaf. The trajectory for this leaf comprises the sequence of nodes spanning from the root to its arrival node. For a data subset $\mathcal{X}_s \subseteq \mathcal{D}$ evaluated across the tree t , let $|v|_{(\mathcal{X}_s, t)}$ indicate the quantity of data points within $\mathcal{X}_s$ that traverse paths intersecting node v . Because the root node is included in every possible path, and exactly one unique path terminates at each individual leaf, it follows that $|\text{head}(t)|_{(\mathcal{X}_s, t)} = |\mathcal{X}_s|$ (meaning all data points initiate their trajectory at the root node).

Complexity of a decision tree

While decision trees can regularly be mapped out through visual diagrams or translated into linear rule sets, such approaches can become unmanageable when the tree exhibits large dimensions—namely in its depth, width, or structural layout, which serve as measures of tree complexity. This perspective, however, overlooks the actual distribution of queries. Specifically, if the major proportion of queries only traverses a straightforward subtree, a structurally intricate tree can still remain practical for the vast majority of cases. The query distribution is explicitly factored in when evaluating decision tree complexity by calculating the mean trace length to the leaves for a representative sample $\mathcal{X}_1 \subseteq \mathcal{D}$, computed as:

$$E_{\mathcal{X}_1}(\text{trace}(t, .)) = \frac{1}{|\mathcal{X}_1|} \sum_{v \in \text{leafs}(t)} |v|_{(\mathcal{X}_1, t)} |\text{trace}(t, v)|. \quad (13)$$

If an appropriate selection for $\mathcal{X}_1$ does not emerge implicitly, it can be constructed by assuming a uniform distribution of samples across the leaves, which yields:

$$E(\text{trace}(t, .)) = \frac{1}{|\text{leafs}(t)|} \sum_{v \in \text{leafs}(t)} |\text{trace}(t, v)|. \quad (14)$$

Reducing the complexity of explanations

Interpretability is closely linked to the cognitive burden imposed on users. We present a strategy to minimize this cognitive load by restricting the LoE architecture to a lower-dimensional feature space. This intervention modifies the optimization process of the LoE while maintaining its predictive accuracy. This restriction likewise carries over to the assignment pipeline and is inherited by the surrogate models. Even though surrogates model intricate, nonlinear decision boundaries, most empirical datasets reside along a lower-dimensional manifold [55], which can be effectively modeled via projections.

A high-dimensional feature space containing a substantial number of attributes frequently degrades the generalization capabilities of machine learning models due to overfitting. Within the training framework of the LoE, the total number of classifiers $|\mathcal{A}|$ scales exponentially. Consequently, multiple surrogates that fail to faithfully represent A can still display strong performance, thereby offering a justification for the assignment without reflecting the true underlying causality. This phenomenon is recognized as the Rashomon set [54].

Feature space reduction

We address the challenges of high dimensionality by mapping the full feature space onto a lower-dimensional subspace. Let $\pi: \mathbb{R}^m \rightarrow \mathbb{R}^r$, specified by $(x) = (x_{i_1}, \dots, x_{i_r})$, represent the projection of $x = (x_1, \dots, x_m)$ onto the r target components $i_1, \dots, i_r$ ($1 \leq i_1 < i_2 < \dots < i_r \leq m$). The distance metric d is substituted by the (pseudo) metric $d_\pi(x, y) = (\pi(x), \pi(y))$, which restricts d to the projection space established by π . Consequently, Eq. 9 is substituted by:

$$\mathcal{A}_n := \left\{ A \mid A(x) := \min_{k \in \{1, \dots, n\}} \{ \arg \min d_\pi(x, \pi(\theta_k)) \} \right. \\ \left. \text{for pairwise different } \theta_k \in \mathbb{R}^m, k = 1, \dots, n \right\}. \quad (15)$$

The adaptation step for the center coordinates of the expert regions is confined to this lower-dimensional projection space, leaving the remaining components unaltered; thus, [Eq. 11](#) is adjusted to:

$$\pi(\theta'_k) = \pi(\theta_k) + \frac{\eta}{|\mathcal{L}|} \sum_{(x,y) \in \mathcal{L}} (\pi(x) - \pi(\theta_k)) p(g_k(x), y), \quad (16)$$

where performance evaluations continue to be executed over the original, unprojected sample set $\mathcal{L}$. Crucially, the surrogates are trained directly within the projected feature space. This constraint limits structural complexity and curtails the Rashomon effect, thereby enhancing the fidelity of the explanations provided by the surrogate. Furthermore, user comprehension is improved because individuals are required to analyze fewer attributes simultaneously.

When decision trees are constructed using a restricted set of attributes, specific variables are more likely to reappear multiple times along a single decision path. These redundant conditions can be condensed into a single unified rule, making explanations easier to interpret. For example, if the conditions $\text{age} < 40$ and $\text{age} > 30$ both occur along a path, these two criteria collapse simply into $\text{age} < 30$. In a similar manner, the combined rules ($\text{age} < 40$ and $\text{age} > 30$) condense into the bounded expression ($\text{age in range } 30 \text{ to } 40$).

Projections obtained from leveraging LoE's properties

The formulation of a projection π depends directly on selecting a subset of attributes, ideally retaining those with the highest significance. For example, variables that demonstrate severe collinearity, offer minor predictive utility, or carry substantial noise can be omitted. Sophisticated implementations draw upon permutation feature importances [\[42\]](#) (Chapter 5.5), SHAP feature importances [\[42\]](#) (Chapter 5.10) built upon Shapley values [\[41\]](#), or direct qualitative assessments of attribute relevance provided by individual experts.

Within the context of LoE (alongside alternative DCS frameworks), the variables that exert the strongest influence on the assignment mechanism simultaneously define their own importance. Every individual expert in an LoE architecture can independently calculate attribute importance rankings, which can subsequently be aggregated into a centralized importance rating. Alternatively, these importance weights can be approximated via a surrogate model h rather than computed directly from the foundational assignment function A . This strategy requires that the designated surrogate class possesses the capability to output feature importance metrics. One potential approach involves introducing a secondary surrogate family $\mathcal{S}_2$ for this specific task; this family does not necessarily need to offer interpretability, but it must be capable of generating feature importance statistics. Extra trees (extremely randomized trees) [\[56\]](#) represent a viable option here. These models are ensembles of decision trees that utilize randomized selections of features and splitting thresholds. Because they bypass the optimization of a formal split criterion, they can be constructed rapidly while delivering robust approximations of feature importance when populated by a sufficiently large ensemble. Furthermore, they escape the structural biases induced by optimization algorithms (such as maximizing the Gini index) during the initial node split formulation.

Feature importance within the LoE framework is partially shaped during its recurrent training cycle, with importance values being extracted at each individual step. These scores can be compiled into moving averages or other time-dependent metrics that reflect long-term attribute relevance.

Concrete calculation of feature importances

We formalize a specific pipeline for computing feature importances, using decision trees as the baseline architectural model. While decision trees represent an intuitive vehicle for this application, alternative models with feature-reporting capabilities can be substituted. When constructing decision trees, the ranking of attributes is intrinsically handled. The importance of any given variable is equated to its splitting utility, which is computed directly from the relative variation of a metric—such as the Gini index—during the tree-building process [57]. In the extra trees setup described previously, this metric is bypassed during the generation of the ensemble itself.

For an LoE ($\mathcal{G}^*$) incorporating an expert ensemble $\{g_1, \dots, g_n\}$ (composed here of decision trees) evaluated on data $\mathcal{L}$, let $\mathcal{L}_k$ signify the specific partition routed to expert g_k (corresponding to the region where $\mathcal{X}_k$). An importance function ϕ assigns each expert the relative significance of every attribute; this constitutes a mapping from the LoE ensemble to an $m-1$ -dimensional simplex ($\phi: \mathcal{G}^* \rightarrow \mathcal{S}_{m-1}$) defined by $\phi: g_k \mapsto (\phi_1(g_k), \dots, \phi_m(g_k))$. The collective feature importance across the entire ensemble is defined as the weighted average of these relative importances across all constituent experts, formulated as:

$$\phi_{\text{experts}}(\mathcal{G}^*) = \sum_{k=1}^n \frac{|\mathcal{X}_k|}{|\mathcal{X}|} \phi(g_k) \in \mathcal{S}_{m-1}. \quad (17)$$

For the routing pipeline, feature importances are calculated through an independent path. Utilizing the secondary surrogate family—configured here as an extra tree ensemble containing 1000 decision trees—a surrogate model is optimized to emulate the assignment function A . An importance function analogous to ϕ is executed on the trained surrogate h to derive the standalone importance of each attribute. The total importance metric then averages these two components using a blending weight α , as follows:

$$\phi_{\text{total}} = \alpha \phi_{\text{experts}}(\mathcal{G}^*) + (1 - \alpha) \tilde{\phi}(h). \quad (18)$$

The total feature importance values are updated dynamically during each step of the LoE algorithm. Rather than utilizing these raw values directly, a temporal average is computed. The time-averaged total feature importance at iteration t , governed by a geometric decay constant β , is expressed as:

$$\Phi_{\text{total}}^{(t)} = \beta \phi_{\text{total}} + (1 - \beta) \Phi_{\text{total}}^{(t-1)}. \quad (19)$$

The projection π designed to condense the dimensionality of the feature space is established by isolating the r features that display the highest cumulative importance scores.

Runtime analysis

To map out the computational complexity of the LoE during its training phase, the analysis relies on the following parameters. First, the key operational variables dictate the overall runtime are summarized. Second, the formal training complexity of the LoE is derived.

- T = number of iterations
- $\mathcal{G}$ = ensemble of experts
- $\mathcal{L}$ = labeled dataset

$C(g, (x, y) \subseteq \mathcal{L})$ = training complexity for model g for data (x, y)

$E(g, (x, y) \subseteq \mathcal{L})$ = evaluation complexity for model g for data (x, y)

$$\mathcal{O}(T \cdot (|\mathcal{G}| \cdot |\mathcal{L}| + \max_{g \in \mathcal{G}} C(g, \mathcal{L}) + \max_{g \in \mathcal{G}} E(g, \mathcal{L}))), \text{ with}$$

$$\begin{aligned} 1. \text{ Assignment data} \rightarrow \text{model:} &= \mathcal{O}(T \cdot |\mathcal{G}| \cdot |\mathcal{L}|) \\ 2. \text{ Expert Training:} &+ \mathcal{O}(T \cdot \max_{g \in \mathcal{G}} C(g, \mathcal{L})) \\ 3. \text{ Model Movement:} &+ \mathcal{O}(T \cdot \max_{g \in \mathcal{G}} E(g, \mathcal{L})) \\ 4. \text{ Performance Evaluation:} &+ \mathcal{O}(T \cdot \max_{g \in \mathcal{G}} E(g, \mathcal{L})) \end{aligned} \quad (20)$$

The upper bound of the iterative cycle in (20) is driven by four sequential operations that repeat across T steps: First, the current routing assignments are evaluated (1). This operation computes the distance matrix between every expert $g \in \mathcal{G}$ and each available data point. Second, the expert optimization and spatial adjustments are performed (2). The individual experts are trained exclusively on their assigned data subsets (i.e., optimized within their local territories), a process restricted by the inherent training complexity of the specific model architectures in $\mathcal{G}$. Third, the optimized models are shifted along the trajectory of their highest-performing feature space positions. Finally, the updated experts execute inference tasks to determine their performance metrics (4), a phase bounded by the prediction complexity of the chosen model types.

The inference complexity of the model across a test sample $\mathcal{X}$ is calculated as follows:

$$|\mathcal{G}| \cdot |\mathcal{X}| + \max_{g \in \mathcal{G}} E(g, \mathcal{X}), \quad (21)$$

This runtime encompasses identifying the target expert (the assignment routine) and subsequently running that specific model for final prediction. Unlike traditional ensemble structures such as random forests or AdaBoost, the training duration scales with the additional multiplier T , causing the training phase to run T times slower. However, during the inference phase, only a single expert model is evaluated instead of the entire pool, avoiding the need to aggregate outputs from all ensemble members. This structure essentially demands $|\mathcal{G}|$ distance updates plus $(g, (x, y))$ for the designated or "closest" model for every $x \in \mathcal{X}$. Consequently, the inference latency of an LoE model scales predictably based on the number of experts, whereas standard ensemble methods scale with the collective, aggregated complexity of all their internal models.

LoE as a rule set learner—RuleLoE

While LoE is not inherently designed as a rule set learner, it can be adapted into an operational equivalent known as RuleLoE by incorporating its surrogate model mechanism. For this integration, we focus primarily on decision trees serving as both experts and surrogates, though alternative rule induction algorithms can be substituted. Explicit decision rules are extracted as logical conjunctions combining two separate stages: (i) the routing pathways that dictate assignment to a specific expert (the surrogate $\rightarrow$ expert assignment path) and (ii) the localized decision pathways within that designated expert leading down to individual leaves. The resulting decision rules are thus structured as conjunctions split into these two components.

Users can manipulate the complexity of the final rule set by adjusting parameters across both dimensions: (i) the structural complexity and approximation fidelity of the surrogate (such as enforcing a maximum depth constraint on a decision tree or applying feature space reductions) and (ii) the total volume and individual complexity of the deployed experts (contrasting a larger ensemble of simpler models against a smaller ensemble of structurally complex models). Furthermore, if a subset of experts exhibits strong global predictive power across the entire sample space, pruning the routing assignment rules can effectively boost model performance, as simplified assignment pathways are more likely to direct queries toward these highly reliable experts.

One-versus-rest surrogate

Rules generated from a singular decision tree surrogate exhibit zero overlap, meaning no two rules can be satisfied simultaneously. Although this mutual exclusivity is advantageous because it prevents prediction ambiguities, it introduces certain structural limitations. Regardless of the query routed through the pipeline, there will always be a degree of partial intersection because every query originates at the root node of the tree and consequently shares at least one initial conditional test. This constraint reduces the flexibility and structural variety of the derived rule set. However, complete rule duplication and excessive partial overlap are equally undesirable since they cause the emerging rules to become overly homogeneous.

In empirical applications, it is beneficial to exchange strict mutual exclusivity for increased rule diversity by actively minimizing partial path overlap. This is achieved by training an independent surrogate model for each available expert. The k -th surrogate learns to model the routing process relative to the k -th expert via a binary framework (a one-versus-rest strategy, or OvR). In practice, each of the k surrogates is provided with a binary-labeled dataset where a target value of 1 denotes $A(x) = k$ (and 0 denotes all other outcomes). The surrogate learns to determine whether the k -th model should be selected to generate a prediction for a given query instance. If the condition is met, the query is assigned to the k -th expert (following its localized decision path); otherwise, no assignment occurs, and that specific expert is ignored. A visual example of this configuration utilizing two surrogates is illustrated in **Figure 5**. It should be noted that applying an OvR strategy within a dual-expert environment is mathematically equivalent to deploying a single surrogate, given that the target labels presented to the two models are direct inverses of each other.

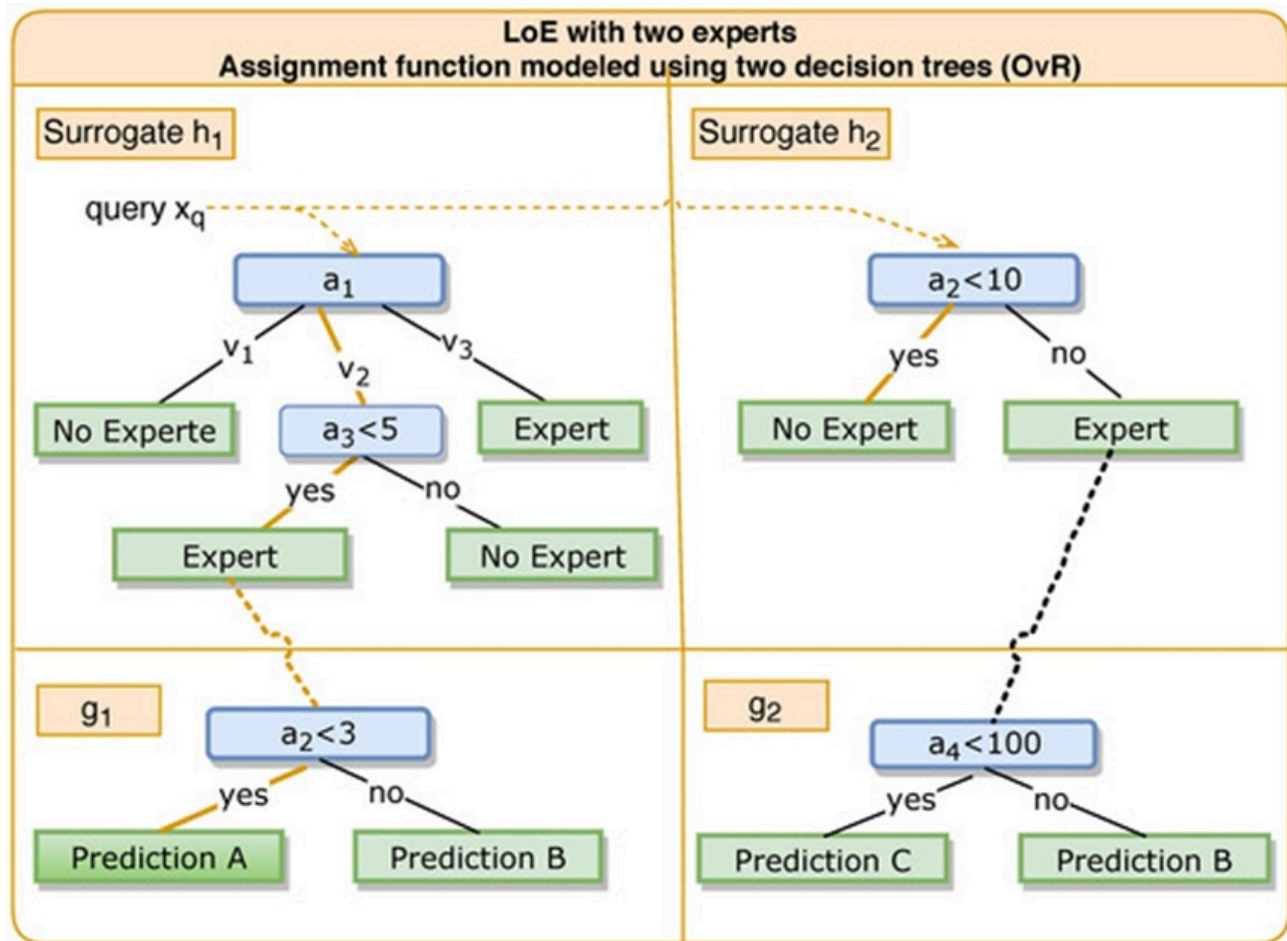


Figure 5. Exemplified LoE instance containing two experts (and) with A being replaced by two decision trees (surrogates). Note that this example is only illustrative as the surrogates would be inverse instances to each other in a 2-expert scenario.

Because the target labels shift for each expert when the ensemble expands beyond two models, every surrogate must be trained on a distinct data distribution. Consequently, each model uncovers unique structural patterns—especially since decision trees exhibit high sensitivity to variations in training inputs—ultimately yielding a more diverse and potentially less complex set of rules. When substituting the original assignment function with surrogate models, the underlying routing logic may not always be perfectly replicated, given that the baseline assignment function can be arbitrarily complex while the surrogate operates under strict complexity constraints or structural limitations. As a result, a given query may trigger either none or multiple expert assignments across the n surrogates.

Query strategy

To resolve the challenges of competing rules and unmapped queries that can occur within decision sets generated by OvR, a query resolution strategy is implemented to mimic human reasoning. When multiple rules compete for an assignment, the rule yielding the highest overall statistical performance is prioritized. Conversely, if an incoming query fails to trigger any existing rule, the closest-fitting rule is selected.

Formally, let $\mathcal{L}_{\text{train}}$ represent the training dataset and r_i denote a specific decision rule (e.g., $r_i \leftarrow \text{age} > 20 \wedge \text{size} < 180 \text{ cm} \Rightarrow \text{positive class}$). If N_i indicates the volume of data points where rule r_i applies, the rule's coverage is formulated as:

$$c_i = N_i |\mathcal{L}|$$

The quantity of covered data points that are correctly mapped by the rule is designated as r_i is N_i^+ , allowing the rule's precision to be defined as:

$$\theta_i = N_i^+ / N_i$$

Incoming queries x_q are mapped to rules based on the following sequential protocol:

1. Rank all available rules by descending precision (resolving ties by prioritizing higher coverage).
2. Assign the query to the covering rule that possesses the lowest rank.
3. If no rule directly covers x_q , identify and return the rule that evaluates the highest proportion of its individual clauses as "true" (partial rule coverage), breaking ties by selecting the rule with superior precision.

This execution sequence, particularly step 3, is feasible only because RuleLoE operates as a multiclass rule learner. This technique cannot be applied to frameworks designed to isolate a single positive class, such as the RIPPER, CN2, or BRG algorithms. While those alternative methods may generate smaller rule sets, they offer fewer descriptive insights into the data.

In summary, the proposed League of Experts algorithm and its structural framework provide an optimized suite of tools tailored for individuals interested in deploying, developing, and benchmark-testing explainable artificial intelligence systems. The foundational code and assets are accessible via the LoE project page and its associated repositories. The project pages for the League of Experts (LoE) ecosystem can be reached through its source repositories at <https://github.com/Mereep/loe>, https://github.com/Mereep/rule_loe, and <https://github.com/Mereep/HDTTree>, accessed on 2 April 2024.

Test Results, Evaluation, and Discussion

The following sections illustrate the practical application of the League of Experts algorithm and its underlying methodology using two distinct empirical datasets from the medical and financial sectors: the UCI breast cancer dataset and the HELOC dataset. Medicine and finance represent two critical domains for interpretable machine learning, as automated predictions carry significant real-world consequences. In medical environments, for instance, a patient's diagnostic path can be evaluated, modified, or clinically justified. In the financial sector, issues of systemic accountability and credit discrimination must be audited when automated systems evaluate loan applications.

Accordingly, two extensive experimental evaluations are executed and discussed below: First, LoE is implemented to analyze the structural impacts of feature space reduction. Second, LoE is converted into its RuleLoE variant, whose interpretability and predictive accuracy are systematically benchmarked against alternative rule induction systems, standard black-box architectures, and widely established ensemble methods. Following this, an in-depth structural inspection of the generated decision rules is conducted using the HELOC dataset. Additionally, a broader series of validation tests are performed across a selection of datasets from the KEEL repository under uniform baseline model settings to provide an overview of performance across varying tasks. A summary of the evaluation datasets is provided below.

- **Example dataset 1: UCI breast cancer** The UCI breast cancer dataset (UCI breast cancer dataset, [https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+\(Diagnostic\)](https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)), accessed on 2 April 2024) [58] comprises 569 instances containing 32 continuous, real-valued attributes. These features describe the geometric characteristics of cell nuclei captured in digitized images of fine-needle aspirates taken from breast masses, covering metrics such as mean radius, worst perimeter, and contour smoothness. The target classification task separates instances into malignant or benign classes.

- **Example dataset 2: HELOC** The HELOC dataset, originally released for the FICO Explainable Machine Learning Challenge (HELOC dataset, <https://community.fico.com/community/xml>, accessed on 2 April 2024) and utilized across multiple explainable AI studies [38, 59, 60], outlines the credit history of 10,549 individual consumers (condensed to 9,872 instances following the removal of duplicate records). The records are defined by 23 metric attributes. Loan default risk is discretized into "good" or "bad" profiles based on whether an individual incurred a payment delinquency lasting 90 days or longer within a 24-month monitoring window.
- **Example dataset 3: KEEL** The KEEL (Knowledge Extraction based on Evolutionary Learning) dataset repository serves as a standardized testing suite designed to "aim at providing to the machine learning researchers a set of benchmarks to analyze the behavior of (...) learning methods" [61]. The KEEL ecosystem contains a wide range of datasets with diverse properties. For this validation work, a specific subset was isolated consisting of medium-scale datasets containing between 5,000 and 10,000 samples (Table 1).

Table 1. Overview of our selected subset of the KEEL dataset with sample sizes between 5000 and 10,000 samples.

Dataset	Entropy of Class Distribution	Number of Categories	Attribute Count	Number of Samples
banana	0.99	2	2	5300
coil2000	0.33	2	85	9822
marketing	3.10	9	13	6876
mushroom	0.96	2	98	5644
optdigits	3.32	10	64	5620
page-blocks	0.63	5	10	5472
phoneme	0.87	2	5	5404
ring	1.00	2	20	7400
satimage	2.48	6	36	6435
texture	3.46	11	40	5500
thyroid	0.45	3	21	7200
twonorm	1.00	2	20	7400

- **Used Software** All codebases and execution frameworks were written in Python (Python project page, <https://www.python.org>, accessed on 2 April 2024, version 3.8). The scikit-learn library (scikit-learn project page, <https://scikit-learn.org>, accessed on 2 April 2024, version 0.22.2.post1) was utilized to set up and run the baseline models for random forests, gradient boosting machines, extra trees, AdaBoost, support vector classifiers (SVC), and multilayer perceptrons (MLP). The external software repositories Wittgenstein (Wittgenstein project page, <https://github.com/imoscovitz/wittgenstein>, accessed on 2 April 2024, version 0.1.6) and AIX360 (AI Explainability 360) [54] (AIX360 project page, <https://github.com/IBM/AIX360>, accessed on 2 April 2024, version 0.2.0) were deployed to implement the RIPPER and BRCA algorithms, respectively.

Feature space reduction

The total number of inputs directly impacts both the designation of the surrogate model h and how the focal points of the expert domains are modified. Here, a decision tree acts as the surrogate model. From an intuitive standpoint, minimizing

the input dimensions streamlines the structural design of the surrogate, which simultaneously enhances its capacity to mimic the allocation mechanism A of the LoE.

For the empirical evaluation, the data collections are split into a 2/3 portion for model training and a 1/3 portion for model testing. The group of surrogates contains decision trees restricted to a maximum depth of 6. The architecture of the LoE is held constant as an ensemble containing $n=3$ experts chosen from the family $\mathcal{G}$ of decision trees that possess a maximum depth of 3. On the other hand, the proportion of features saved by the projections π is shifted between 10% and 100% using steps of 5% ($r=n \times p$ with $p=0.1, 0.15, \dots, 1$). For every distinct value of r , the LoE undergoes training and testing across 20 separate iterations. These iterations utilize identical data partitions that are applied consistently across each step of r . The evaluation records the training and testing accuracies, the mean path distance of the surrogate decision trees h , and the allocation success metric θ_{ass} (which indicates how closely the surrogate mirrors the LoE allocation function).

Input importance is derived using Eq. 19 with assigned weights of $\alpha = 0.5$ (Eq. 18) and a geometric decay rate of $\beta = 0.3$ (Eq. 19). These specific parameters were not tuned for optimization. Additionally, the decision paths of the surrogates were not simplified through any consolidation techniques.

Test Results

When applied to the UCI breast cancer dataset, the LoE framework utilizing three experts demonstrates a slight tendency to overfit the training partitions. This trend is evident when comparing the training scores against the test scores (Figure 6, contrasting C and D). The training phase yields a success rate near 100%, whereas the testing phase drops closer to 94%. This divergence indicates that dropping an expert or enforcing stricter constraints on them would likely improve generalization.

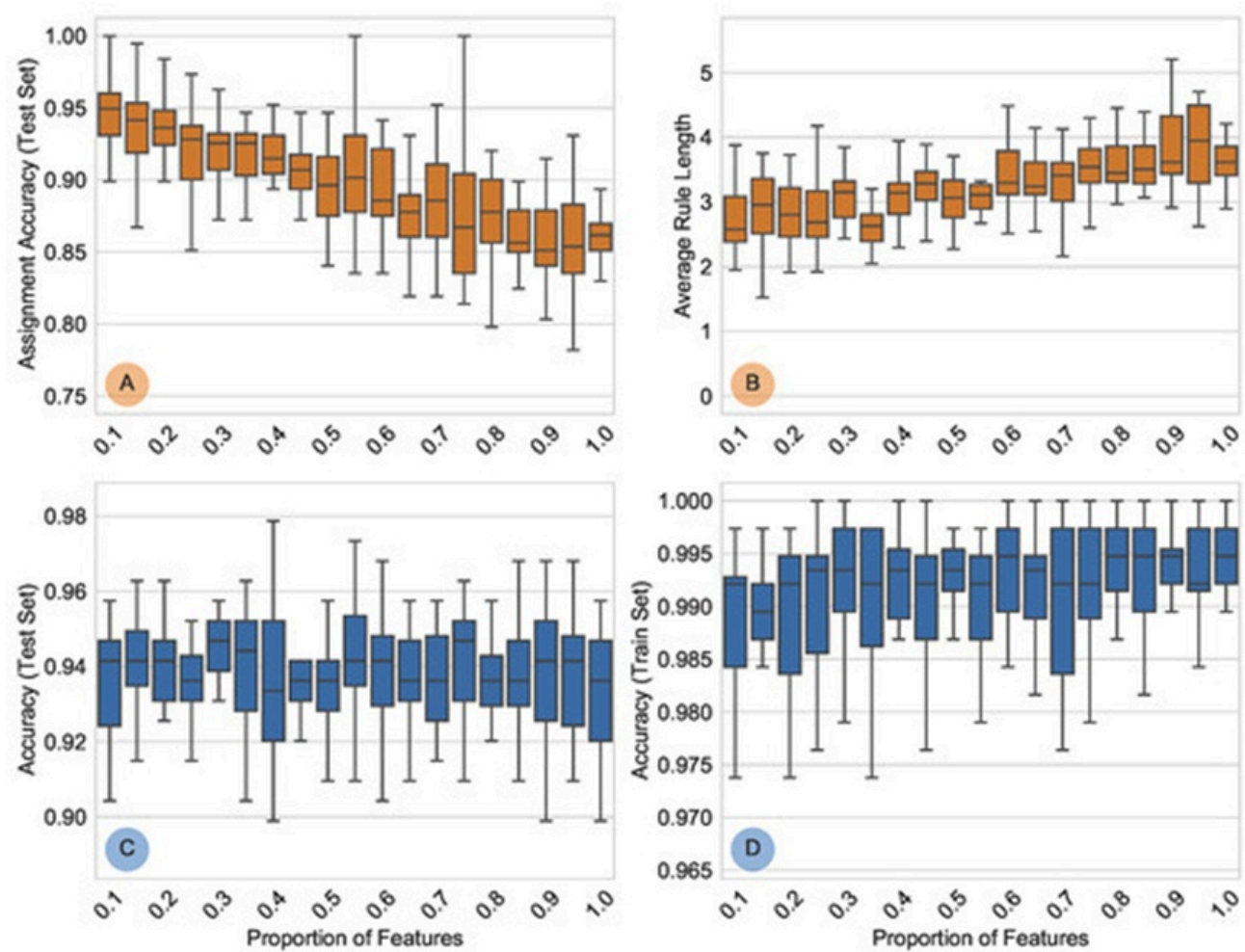


Figure 6. UCI breast cancer dataset results. Shown are paired box plots with their proportions of retained features over 20 runs for (A) assignment accuracy (p_{ass}), measured on test data, (B) average rule/path length, (C) accuracy for the test data, and (D) accuracy for the train data.

Restricting the feature space heavily impacts both the structural complexity and the overall fidelity of the surrogate model. As displayed in **Figures 6A, 6B**, the mean rule length (the weighted average of the path depth) shows a pronounced positive correlation with the volume of retained features (Pearson's χ^2 test: $\chi^2=0.48$, $p<0.01$). Concurrently, it displays a strong negative correlation with the allocation accuracy (the degree to which the surrogate matches $\mathcal{A}$; $\chi^2=-0.62$, $p<0.01$).

We observe parallel trends when evaluating the FICO data (**Figure 7**). Overall accuracy metrics are lower for both the training and testing sets, yet the underlying pattern of overfitting resembles the behavior seen with the UCI dataset (**Figures 7C, 7D**). However, eliminating features causes an even more dramatic shift in this scenario: the rule length shares an even stronger positive correlation with the number of attributes ($\chi^2=0.57$, $p<0.01$). Similarly, a more severe negative correlation with the allocation accuracy is present ($\chi^2=-0.94$, $p<0.01$). These powerful statistical relationships highlight how vital it is to identify the ideal target dimensionality when compressing the feature space. When more than 35% of the attributes are preserved, the mean rule length rapidly climbs toward the maximum ceiling of 6 dictated by the surrogate model family. This pattern implies that the extracted decision rules would expand in complexity if they were not artificially capped, meaning the surrogate would overfit $\mathcal{A}$.

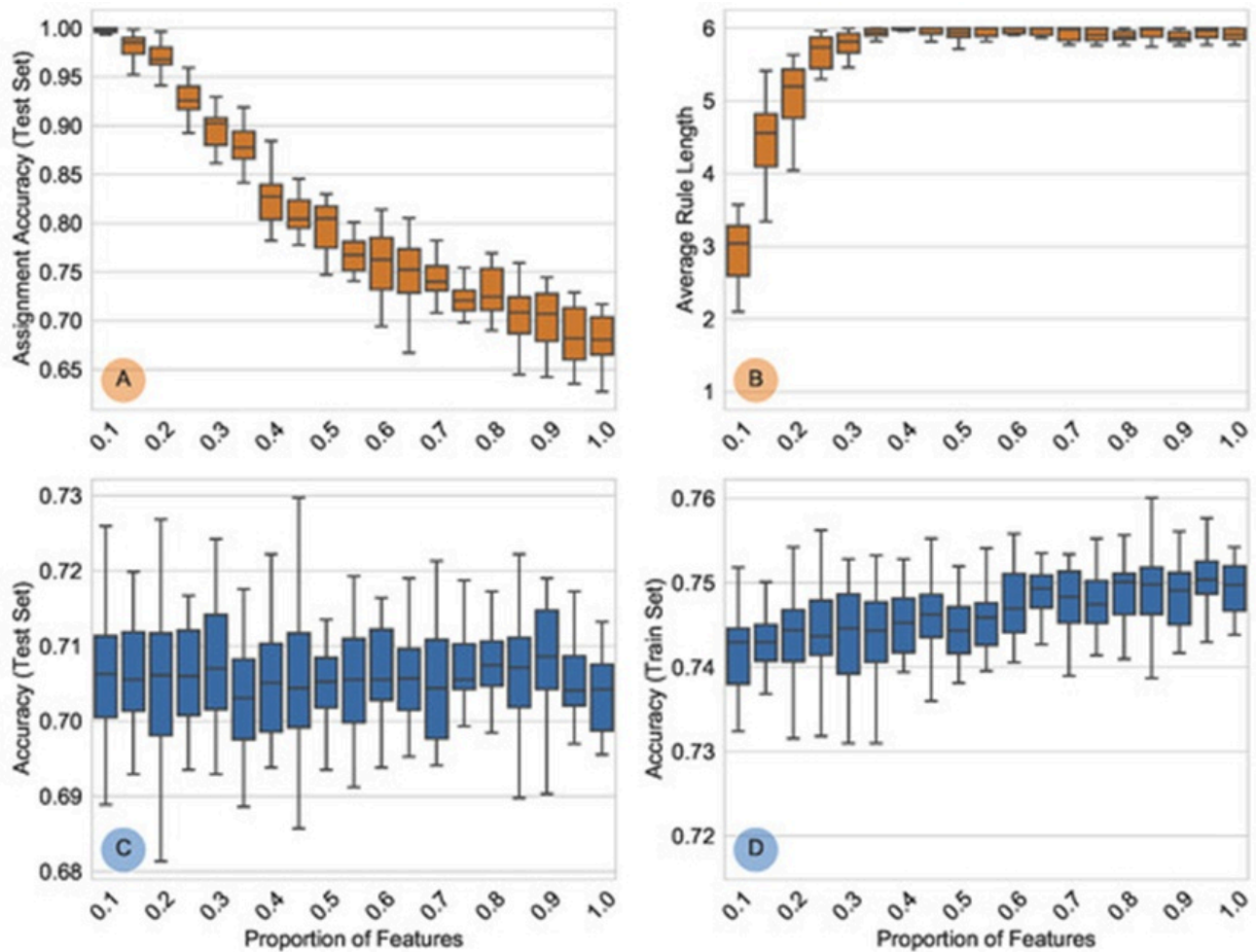


Figure 7. FICO dataset results. Shown are paired box plots with their proportions of retained features over 20 runs for (A) assignment accuracy (p_{ass}), measured on test data, (B) average rule/path length, (C) accuracy for the test data, and (D) accuracy for the train data.

Rule set generation with RuleLoE

The performance attributes of RuleLoE regarding (i) interpretability (mean rule length, $\emptyset$ rules, and # rules), (ii) fidelity (rule coverage), and (iii) classification accuracy are measured against alternative rule-induction frameworks. Specifically, comparisons are made with (1) RIPPER [36] (a model that remains a state-of-the-art benchmark despite its historical background; see [46] (p. 52) and [38]), (2) Boolean decision rules via column generation (BRCG) [38], and (3) standard CART decision trees. To evaluate raw accuracy, RuleLoE is also benchmarked against two opaque black box models, SVC and MLP, which use standard parameter settings unless specified otherwise.

Given that LoE is structured as a type of ensemble learning, we also compare it against a variety of well-known alternative ensemble algorithms—specifically random forest, gradient boosting, extra trees, and AdaBoost—to offer a broader context. Because tree-based ensemble frameworks rely on concepts of ensemble capacity and maximum tree depth that match the logic of LoE, we tested two configurations for each: a default version and a matched version. The default versions use the standard hyperparameter configurations of the algorithm. The matched versions alter those default settings by locking the ensemble capacity and maximum tree depth to correspond exactly with the configurations

used in LoE. The baseline ensemble setups, organized as pairs of (ensemble size / max tree depth), are configured as follows:

- Random forest: 100/ ∞
- Gradient boosting: 100/3
- Extra trees: 100/ ∞
- AdaBoost: 50/1

At this juncture, it should be clarified that standard AdaBoost only generates decision stumps, which are single-level trees with a maximum depth of one. Even though these baseline ensemble models are built from decision trees, extracting clear explanations from them is difficult because their individual outputs must be combined during the final prediction phase. Especially under default hyperparameter setups, these algorithms build massive ensembles that essentially act as uninterpretable black boxes.

UCI breast cancer dataset

The initial test configuration used an LoE ensemble composed of $n=9$ decision trees with a maximum depth of 1 level, alongside surrogate trees capped at a maximum depth of 2, with the feature space compressed down to $r=11$ attributes. Following the training process, a single expert that commanded a very small territory was manually pruned from the system.

The structural complexity of the generated rules fluctuates widely across the different models, revealing no uniform pattern. RIPPER and BRCG isolate a single target class (either positive or negative) to construct their rules, whereas LoE and CART account for both classes simultaneously, modifying both their structural complexity and performance. Consequently, RIPPER and BRCG demonstrate lower overall coverage because they map fewer data points to any given rule. In contrast, the coverage for standard decision trees (CART) reaches 100% because every possible inquiry can be traced down a valid path to a leaf node. In this evaluation, RuleLoE matches the effectiveness of the alternative frameworks, including the black box configurations. Additionally, the transparent glass box alternatives were surpassed in terms of testing accuracy (Table 2). None of the tested models sweep every evaluation category. In this scenario, there is no clear justification for choosing an uninterpretable black box model over a transparent glass box alternative.

Table 2. UCI breast cancer results. Shown are the number of rules (# rules), their average length (lvarnothing rules), rule coverage, and (test/training) accuracies. *Italic values for RIPPER and BRCG specify the learned positive class.* Our found hyperparameters in Python notation using grid search are `ccp_alpha=0.01` (CART, post pruning); `hidden_layer_sizes=(200,)` and `max_iter=500` (MLP).

Method	Configuration / Target Class	Classification Accuracy (%)	Rule Coverage (%)	Average Rule Length	Total Rules
RuleLoE	—	94.22 / 96.28	96.06 / 95.21	1.70	10
RIPPER	malignant	95.54 / 95.21	39.90 / 36.17	1.56	9
RIPPER	benign	93.20 / 89.89	61.41 / 61.70	1.53	13
BRCG	malignant	98.95 / 95.21	37.00 / 35.11	2.20	5
BRCG	benign	99.48 / 92.02	61.42 / 62.77	2.80	5
CART	—	97.11 / 95.74	100.00 / 100.00	3.00	7

Random Forest	default	100.00 / 96.28	–	–	–
Random Forest	adjusted	94.75 / 92.02	–	–	–
Gradient Boosting	default	100.00 / 95.21	–	–	–
Gradient Boosting	adjusted	94.23 / 94.15	–	–	–
Extra Trees	default	100.00 / 97.87	–	–	–
Extra Trees	adjusted	76.64 / 78.19	–	–	–
AdaBoost	default	100.00 / 95.21	–	–	–
AdaBoost	adjusted	98.95 / 96.81	–	–	–
SVC	—	90.55 / 95.21	–	–	–
MLP	—	94.22 / 95.21	–	–	–

FICO dataset

The initial configuration consisted of an LoE ensemble composed of n=4 experts restricted to a maximum depth of 1, paired with surrogate decision trees capped at a maximum depth of 2 that preserved r=8 features. Due to an excessively high concentration of data points coupled with very poor purity (i.e., a highly skewed class distribution) at one of its terminal nodes, a single expert was manually expanded to a depth of two levels.

While RIPPER and CART extract an extensive volume of rules, they fail to deliver superior predictive accuracy (Table 3); in fact, RIPPER exhibits marginally lower performance. Furthermore, the rules generated by CART are highly intricate. Conversely, BRCG (utilizing bad as the positive class) identifies exactly one brief rule (Predict y=bad if: ExternalRiskEstimate < 73) that delivers strong predictive performance (Table 3). This aligns with expectations, as ExternalRiskEstimate serves as a highly meaningful predictor for creditworthiness; it aggregates various risk indicators to compute an overall risk score through an undisclosed functional relationship. Consequently, the straightforward classification rule produced by BRCG relies on a rigid threshold that remains opaque to external observers. In contrast, RuleLoE yields a different set of pathways, resulting in the subsequent rule set (Figure 8).

Predict y=good if one of the following applies:

- (1): ExternalRiskEstimate ≥ 76
- (2): AverageMInFile $\geq 54 \wedge$ MSinceMostRecentInqexcl7days $\geq 1 \wedge$
ExternalRiskEstimate $\in [70, 76[$

Predict y=bad if one of the following applies:

- (3): ExternalRiskEstimate $< 69 \wedge$ ExternalRiskEstimate $\neq 68$
- (4): AverageMInFile $\geq 54 \wedge$ MSinceMostRecentInqexcl7days $< 1 \wedge$
ExternalRiskEstimate $\in [70, 76[$
- (5): ExternalRiskEstimate INSIDE $[68, \dots, 70[$
- (6): AverageMInFile $< 54 \vee$ ExternalRiskEstimate $\in [70, 76[$

Figure 8. FICO dataset results. Shown are the obtained rules via RuleLoE.

Table 3. FICO dataset results. Shown are the number of rules (# rules), their average length ($\varnothing$ rules), rule coverage, and (test / training) accuracies. *Italic values for RIPPER and BRCG specify the learned positive class.* Our found hyperparameters using grid search are ccp_alpha=0.001 (CART, post pruning); hidden_layer_sizes=(8,8) (MLP).

Technique	Variant / Target Label	Prediction Accuracy (%)	Coverage of Rules (%)	Mean Rule Size	Number of Rules
RuleLoE	—	72.04 / 71.73	100.00 / 100.00	2.00	6
RIPPER	bad	73.95 / 70.03	50.44 / 49.66	2.50	18
RIPPER	good	73.95 / 71.49	47.05 / 47.39	2.40	25
BRCG	bad	71.12 / 70.38	56.01 / 56.29	1.00	1
BRCG	good	72.35 / 70.53	48.50 / 48.22	3.00	2
CART	—	74.05 / 71.24	100.00 / 100.00	4.86	21
Random Forest	default	100.00 / 72.41	—	—	—
Random Forest	adjusted	100.00 / 72.41	—	—	—
Gradient Boosting	default	62.25 / 62.83	—	—	—
Gradient Boosting	adjusted	76.70 / 73.20	—	—	—
Extra Trees	default	71.12 / 70.38	—	—	—
Extra Trees	adjusted	62.49 / 62.86	—	—	—
AdaBoost	default	74.89 / 72.71	—	—	—
AdaBoost	adjusted	71.05 / 69.28	—	—	—
SVC	—	73.20 / 72.41	—	—	—

MLP	—	71.58 / 70.17	—	—	—
-----	---	---------------	---	---	---

Certain pathways from this set could undergo further consolidation, such as rules (3) and (5). Nonetheless, these classification pathways inherently incorporate the same opaque metric, `ExternalRiskEstimate`. However, unlike the binary thresholding of BRCG, this model uncovers additional contextual information. Whenever the metric `ExternalRiskEstimate` fails to provide an unambiguous signal for a favorable or unfavorable credit rating (≥ 76 or < 70 , respectively), the case is treated as borderline, and the final assessment shifts to two fully transparent attributes. Specifically, applicants are designated as having a favorable credit score if their history spans a mean of more than 54 months in system records and their most recent credit inquiry occurred more than 1 month prior. Formally, a credit profile is deemed favorable if `AverageMInFile $\geq$ 54 ^ MSinceMostRecentInqexcl7days $<$ 1`. In all other instances, the profile is classified as unfavorable.

These more informative decision paths produced by RuleLoE exhibit lower structural complexity than the alternative techniques, with the sole exception of BRCG configured for the positive class bad, which occasionally yields marginal performance gains (Table 2). As a consequence, opaque black box models do not generally outperform transparent glass box frameworks regarding accuracy within our tested scenarios; notably, the SVC achieves only a negligible edge over the remaining algorithms. Weighing all evaluated criteria, RuleLoE demonstrates superior overall utility compared to the alternative approaches, as it facilitates active knowledge discovery by generating straightforward, highly interpretable decision pathways that maximize explainability.

The KEEL dataset

For the entire suite of evaluations involving the KEEL archive, we deployed an LoE ensemble featuring three distinct experts restricted to a maximum depth of 2. This highly constrained architecture was intentionally chosen to prioritize the extraction of compact, easily interpretable rule architectures, directly supporting the core objective of generating explanation-driven models. The underlying data were partitioned using a standard 2/3 split for training and testing stages. To establish a comparative baseline, a random forest model was evaluated under identical constraints regarding ensemble volume and maximum tree depth (Table 4).

Even without implementing dataset-specific hyperparameter tuning, the LoE achieves a thoroughly respectable level of performance. A few of our experimental trials reveal a low allocation accuracy (specifically on the marketing and texture data segments), which subsequently penalizes the downstream performance of RuleLoE. This degradation can likely be traced to an inadequate depth configuration within the corresponding allocation trees or an insufficient degree of feature reduction. The data segments corresponding to marketing, optdigits, and texture—which demonstrate suboptimal overall metrics—are characterized by a high number of distinct classes (Table 1). These multi-class landscapes cannot be adequately mapped by decision trees restricted to a depth of only two levels, naturally introducing higher misclassification rates. On average, however, both LoE and RuleLoE surpass the restricted random forest architecture in terms of predictive accuracy, highlighting the structural advantages of LoE for generating efficient, highly interpretable rule-based systems under strict architectural constraints. Consequently, future research should investigate these specific performance variations in greater depth, incorporating a broader array of baseline models.

Table 4. Test results for the KEEL dataset, comparing the accuracy of LoE, a derived RuleLoE, and a random forest classifier as pairs of train/test accuracies. Additionally, the number of rules and their average length is reported. Best average results are highlighted in bold.

Dataset	Extracted Rules (Average Length)	Assignment Precision (%)	RuleLoE Performance (%)	LoE Performance (%)	Random Forest Performance (%)

banana	10 (2.20)	99.35 / 99.31	83.55 / 82.68	83.75 / 83.02	64.74 / 63.87
coil2000	4 (2.75)	99.83 / 99.91	94.01 / 94.08	94.18 / 93.99	94.01 / 94.08
marketing	1 (2.00)	28.29 / 26.87	18.35 / 18.06	32.48 / 32.91	27.25 / 27.97
mushroom	8 (3.62)	99.07 / 99.52	98.92 / 99.41	99.84 / 99.89	90.08 / 89.43
optdigits	8 (3.88)	57.24 / 58.22	52.88 / 51.27	65.34 / 63.83	55.35 / 56.93
page-blocks	5 (2.80)	98.01 / 98.28	94.27 / 93.58	95.69 / 94.57	93.97 / 93.47
phoneme	9 (3.11)	90.08 / 89.69	76.69 / 76.35	79.53 / 79.93	75.58 / 76.18
ring	9 (2.44)	84.79 / 84.77	73.26 / 71.25	76.32 / 74.12	76.38 / 74.53
satimage	10 (3.40)	83.86 / 84.42	77.38 / 78.30	82.16 / 83.57	71.12 / 71.05
texture	9 (3.33)	62.61 / 63.86	60.22 / 59.89	70.91 / 70.19	36.28 / 35.32
thyroid	5 (1.60)	98.13 / 98.40	98.11 / 98.36	97.57 / 98.15	94.63 / 93.98
twonorm	3 (3.00)	75.47 / 75.23	74.16 / 75.39	94.61 / 94.27	80.33 / 79.81
Mean Across Datasets	6.75 (2.84)	81.39 / 81.54	75.15 / 74.88	81.03 / 80.70	71.64 / 71.38

Conclusion

In this study, we introduced the League of Experts (LoE) framework as a novel architectural paradigm within the fields of explainable artificial intelligence and explainable machine learning (XAI/XML). The LoE represents a unique class of ensemble learning that integrates surrogate models specifically designed to maximize model interpretability and enable seamless human-in-the-loop interaction. Working alongside this architecture is RuleLoE, a specialized rule-induction method derived from the parent ensemble. By selecting individual ensemble components—the experts—exclusively from transparent glass box models, the broader LoE framework functions inherently as a glass box system. As our experiments demonstrate, this approach maintains competitive predictive performance when evaluated against established glass box algorithms and opaque black box models alike. Crucially, feature space compression mechanisms can be natively embedded directly into the LoE training sequence, thereby reducing the structural complexity of the resulting explanations. This offers a stark contrast to conventional methodologies that must navigate through high-dimensional feature spaces without such safeguards. Moving forward, we intend to focus on hyperparameter optimization, targeting both the establishment of robust baseline defaults and the automated contextual tuning of ensemble components based on specific dataset characteristics and runtime training diagnostics.

As a result, numerous pathways for refining both LoE and RuleLoE remain to be fully investigated concerning operational usability, predictive throughput, and explanation visualization. While our current experimental validations utilized decision trees as the core building blocks for the LoE, the framework accommodates a wide variety of base learners. Incorporating alternative model classes could enable the architecture to capture distinct attribute distributions more effectively, while blending heterogeneous models might map highly intricate data distributions while fully preserving system explainability. To ensure the practical utility of these methodologies, comprehensive options for direct user interaction (such as pruning specific nodes from decision paths, establishing whitelists or blacklists for certain attributes relative to designated experts, or manually discarding individual pathways from RuleLoE) must be built into intuitive user interfaces. The development of these tools will be guided by targeted user experience studies. An interface of this caliber would ideally streamline the injection of domain-specific human expertise directly into the active machine learning pipeline. Structured user interaction has the clear potential to minimize latent algorithmic biases and enhance causal decision mining. Accordingly, this initial introduction of the LoE framework serves as a foundational benchmark for subsequent advancements.

Abbreviations

The following abbreviations are used in this manuscript:

ANN	artificial neural network
BRCG	Boolean decision rules via column generation
CART	classification and regression trees algorithm
DCS	dynamic classifier selection
DES	dynamic ensemble selection
DS	dynamic selection
DSEL	split of training data used in some MCS algorithms
ICE	individual conditional expectation plots
<i>k</i> -NN	<i>k</i> -nearest neighbors
LoE	League of Experts
MCS	multiple classifier system
ML	machine Learning
MLP	multilayer perceptron
OvR	one-versus-rest
PDP	partial dependence plots
RuleLoE	League of Experts rule set learner
SVC	support vector classifier
SVM	support vector machine
TreeSHAP	decision-tree-based SHAP
XAI	explainable artificial intelligence
XML	explainable machine learning

Acknowledgements

None

Conflict of interest

None

Financial support

This study was carried out in the group of junior scientists “Agile Publika”, which is funded by the European Social Fund (ESF) and the Free State of Saxony, Germany. The work of Kristan Alexander Schneider was partially performed within the DFG project “SCH 1480/2-1”.

Ethics statement

None

Received: 05 Sep 2023

Revised: 29 Oct 2023

Accepted: 22 Nov 2023

Published online: 15 March 2024

Rights and permissions

Open Access The author(s) retain copyright. This article is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License. It may be shared and adapted for non-commercial purposes with appropriate attribution, an indication of changes, and distribution of adaptations under the same license. Third-party material may be subject to separate terms identified in its credit line. View the license at <https://creativecommons.org/licenses/by-nc-sa/4.0/>.

References

- Maurer M, Gerdes JC, Lenz B, Winner H, editors. Autonomous driving: technical, legal and social aspects. Berlin: Springer; 2016. <https://doi.org/10.1007/978-3-662-48847-8>.
- Haynes RB, Wilczynski NL. Effects of computerized clinical decision support systems on practitioner performance and patient outcomes: methods of a decision-maker-researcher partnership systematic review. *Implement Sci*. 2010;5:12. <https://doi.org/10.1186/1748-5908-5-12>.
- Bennett Moses L, Chan J. Algorithmic prediction in policing: assumptions, evaluation, and accountability. *Polic Soc*. 2018;28:806-22. <https://doi.org/10.1080/10439463.2018.1439225>.
- Goodman B, Flaxman S. European Union regulations on algorithmic decision-making and a "right to explanation". *AI Mag*. 2017;38(3):50-7. <https://doi.org/10.1609/aimag.v38i3.2741>.
- Deeks A. The judicial demand for explainable artificial intelligence. *Columbia Law Rev*. 2019;119:1829-50.
- Doshi-Velez F, Kortz M, Budish R, Bavitz C, Gershman S, O'Brien D, et al. Accountability of AI under the law: the role of explanation [Internet]. arXiv. 2017 [cited 2026 Sep 8]. Available from: <https://arxiv.org/abs/1711.01134>
- Rudin C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nat Mach Intell*. 2019;1:206-15. <https://doi.org/10.1038/s42256-019-0048-x>.
- Swartout W, Paris C, Moore J. Explanations in knowledge systems: design for explainable expert systems. *IEEE Expert*. 1991;6(3):58-64.
- Paris CL. Generation and explanation: building an explanation facility for the explainable expert systems framework. In: *Natural language generation in artificial intelligence and computational linguistics*. Berlin: Springer; 1991. p.49-82.

Confalonieri R, Coba L, Wagner B, Besold TR. A historical perspective of explainable artificial intelligence. *WIREs Data Min Knowl Discov*. 2021;11:e1391.
<https://doi.org/10.1002/widm.1391>.

Longo L, Brcic M, Cabitza F, Choi J, Confalonieri R, Del Ser J, et al. Explainable artificial intelligence (XAI) 2.0: a manifesto of open challenges and interdisciplinary research directions [Internet]. *arXiv*. 2023 [cited 2026 Sep 8]. Available from:
<https://arxiv.org/abs/2310.19775>

Markus AF, Kors JA, Rijnbeek PR. The role of explainability in creating trustworthy artificial intelligence for health care: a comprehensive survey of the terminology, design choices, and evaluation strategies. *J Biomed Inform*. 2021;113:103655.
<https://doi.org/10.1016/j.jbi.2020.103655>.

Zini JE, Awad M. On the explainability of natural language processing deep models. *ACM Comput Surv*. 2022;55(5):1-31.
<https://doi.org/10.1145/3530188>.

Band SS, Yarahmadi A, Hsu CC, Biyari M, Sookhak M, Ameri R, et al. Application of explainable artificial intelligence in medical health: a systematic review of interpretability methods. *Inform Med Unlocked*. 2023;40:101286.
<https://doi.org/10.1016/j.imu.2023.101286>.

Ali S, Abuhmed T, El-Sappagh S, Muhammad K, Alonso-Moral JM, Confalonieri R, et al. Explainable artificial intelligence (XAI): what we know and what is left to attain trustworthy artificial intelligence. *Inf Fusion*. 2023;99:101805.
<https://doi.org/10.1016/j.inffus.2023.101805>.

Crook B, Schlüter M, Speith T. Revisiting the performance-explainability trade-off in explainable artificial intelligence (XAI). In: *Proceedings of the 2023 IEEE 31st International Requirements Engineering Conference Workshops (REW)*; 2023; Hannover, Germany. Piscataway (NJ): IEEE; 2023. p.316-24.

Hastie T, Tibshirani R, Friedman J. *The elements of statistical learning: data mining, inference, and prediction*. 2nd ed. New York (NY): Springer; 2009.
<https://doi.org/10.1007/978-0-387-84858-7>.

Cruz RMO, Sabourin R, Cavalcanti GDC. Dynamic classifier selection: recent advances and perspectives. *Inf Fusion*. 2018;41:195-216.
<https://doi.org/10.1016/j.inffus.2017.09.010>.

Breiman L. Bagging predictors. *Mach Learn*. 1996;24:123-40.
<https://doi.org/10.1007/BF00058655>.

Guidotti R, Monreale A, Ruggieri S, Turini F, Giannotti F, Pedreschi D. A survey of methods for explaining black box models. *ACM Comput Surv*. 2018;51(5):93.
<https://doi.org/10.1145/3236009>.

Adadi A, Berrada M. Peeking inside the black-box: a survey on explainable artificial intelligence (XAI). *IEEE Access*. 2018;6:52138-60.
<https://doi.org/10.1109/ACCESS.2018.2870052>.

Loyola-Gonzalez O. Black-box vs. white-box: understanding their advantages and weaknesses from a practical point of view. *IEEE Access*. 2019;7:154096-113.
<https://doi.org/10.1109/ACCESS.2019.2949286>.

Bodria F, Giannotti F, Guidotti R, Naretto F, Pedreschi D, Rinzivillo S. Benchmarking and survey of explanation methods for black box models. *Data Min Knowl Disc*. 2023;37:1719-78.
<https://doi.org/10.1007/s10618-023-00933-9>.

LeCun Y, Bengio Y, Hinton G. Deep learning. *Nature*. 2015;521:436-44.
<https://doi.org/10.1038/nature14539>.

Canziani A, Paszke A, Culurciello E. An analysis of deep neural network models for practical applications [Internet]. arXiv. 2016 [cited 2026 Sep 8]. Available from: <https://arxiv.org/abs/1605.07678>

Drucker H, Burges CJC, Kaufman L, Smola AJ, Vapnik V. Support vector regression machines. In: *Advances in neural information processing systems* 9; 1996. p.155-61.

Freund Y, Schapire RE. A decision-theoretic generalization of on-line learning and an application to boosting. *J Comput Syst Sci*. 1997;55:119-39.
<https://doi.org/10.1006/jcss.1997.1504>.

Breiman L. Random forests. *Mach Learn*. 2001;45:5-32.
<https://doi.org/10.1023/A:1010933404324>.

Breiman L, Friedman JH, Olshen RA, Stone CJ. *Classification and regression trees*. Boca Raton (FL): Routledge; 2017.

Das A, Rad P. Opportunities and challenges in explainable artificial intelligence (XAI): a survey [Internet]. arXiv. 2020 [cited 2026 Sep 8]. Available from: <https://arxiv.org/abs/2006.11371>

Hassija V, Chamola V, Mahapatra A, Singal A, Goel D, Huang K, et al. Interpreting black-box models: a review on explainable artificial intelligence. *Cogn Comput*. 2024;16:45-74.
<https://doi.org/10.1007/s12559-023-10179-8>.

Altman NS. An introduction to kernel and nearest-neighbor nonparametric regression. *Am Stat*. 1992;46:175-85.
<https://doi.org/10.1080/00031305.1992.10475879>.

Friedman N, Geiger D, Goldszmidt M. Bayesian network classifiers. *Mach Learn*. 1997;29:131-63.
<https://doi.org/10.1023/A:1007465528199>.

Lakkaraju H, Bach SH, Leskovec J. Interpretable decision sets: a joint framework for description and prediction. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*; 2016; San Francisco, CA. New York (NY): ACM; 2016. p.1675-84.
<https://doi.org/10.1145/2939672.2939874>.

Clark P, Niblett T. The CN2 induction algorithm. *Mach Learn*. 1989;3:261-83.
<https://doi.org/10.1007/BF00116835>.

Cohen WW. Fast effective rule induction. In: *Proceedings of the Twelfth International Conference on Machine Learning*; 1995; Tahoe City, CA. San Francisco (CA): Morgan Kaufmann; 1995. p.115-23.
<https://doi.org/10.1016/B978-1-55860-377-6.50023-2>.

Yang H, Rudin C, Seltzer M. Scalable Bayesian rule lists [Internet]. arXiv. 2016 [cited 2026 Sep 8]. Available from: <https://arxiv.org/abs/1602.08610>

Dash S, Gunluk O, Wei D. Boolean decision rules via column generation. In: Bengio S, Wallach H, Larochelle H, Grauman K, Cesa-Bianchi N, Garnett R, editors. *Advances in neural information processing systems* 31. Red Hook (NY): Curran Associates; 2018. p.4655-65.

Ribeiro MT, Singh S, Guestrin C. Why should I trust you? Explaining the predictions of any classifier [Internet]. arXiv. 2016 [cited 2026 Sep 8]. Available from: <https://arxiv.org/abs/1602.04938>

Ribeiro MT, Singh S, Guestrin C. Anchors: high-precision model-agnostic explanations. In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*; 2018; New Orleans, LA. Palo Alto (CA): AAAI Press; 2018. 32(1).

<https://doi.org/10.1609/aaai.v32i1.11491>.

Lundberg SM, Lee S. A unified approach to interpreting model predictions. In: Guyon I, Luxburg UV, Bengio S, Wallach HM, Fergus R, Vishwanathan SVN, et al., editors. *Advances in neural information processing systems* 30. Red Hook (NY): Curran Associates; 2017. p.4768-77.

Molnar C. *Interpretable machine learning: a guide for making black box models interpretable* [Internet]. 2019 [cited 2026 Sep 8]. Available from: <https://christophm.github.io/interpretable-ml-book/>

Lundberg SM, Erion GG, Lee SI. Consistent individualized feature attribution for tree ensembles [Internet]. arXiv. 2018 [cited 2026 Sep 8]. Available from: <https://arxiv.org/abs/1802.03888>

Buhrmester V, Münch D, Arens M. Analysis of explainers of black box deep neural networks for computer vision: a survey [Internet]. arXiv. 2019 [cited 2026 Sep 8]. Available from: <https://arxiv.org/abs/1911.12116>

Arrieta AB, Díaz-Rodríguez N, Del Ser J, Benetot A, Tabik S, Barbado A, et al. Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI. *Inf Fusion*. 2020;58:82-115.
<https://doi.org/10.1016/j.inffus.2019.12.012>.

Fürnkranz J, Gamberger D, Lavrač N. *Foundations of rule learning*. Berlin: Springer; 2012.
<https://doi.org/10.1007/978-3-540-75197-7>.

Gunning D. DARPA's explainable artificial intelligence (XAI) program. In: *Proceedings of the 24th International Conference on Intelligent User Interfaces*; 2019; Los Angeles, CA. New York (NY): ACM; 2019. p.ii.
<https://doi.org/10.1145/3301275.3308446>.

Kotsiantis SB. Decision trees: a recent overview. *Artif Intell Rev*. 2013;39:261-83.
<https://doi.org/10.1007/s10462-011-9272-4>.

Freitas AA. Comprehensible classification models. *ACM SIGKDD Explor Newsl*. 2014;15(1):1-10.
<https://doi.org/10.1145/2594473.2594475>.

Alizadeh R, Allen JK, Mistree F. Managing computational complexity using surrogate models: a critical review. *Res Eng Des*. 2020;31:275-98.
<https://doi.org/10.1007/s00163-020-00336-5>.

Heider KG. The Rashomon effect: when ethnographers disagree. *Am Anthropol*. 1988;90:73-81.
<https://doi.org/10.1525/aa.1988.90.1.02a00050>.

Dong X, Yu Z, Cao W, Shi Y, Ma Q. A survey on ensemble learning. *Front Comput Sci*. 2020;14:241-58.
<https://doi.org/10.1007/s11704-019-8208-z>.

Mienye ID, Sun Y. A survey of ensemble learning: concepts, algorithms, applications, and prospects. *IEEE Access*. 2022;10:99129-49.
<https://doi.org/10.1109/ACCESS.2022.3207287>.

Arya V, Bellamy RKE, Chen PY, Dhurandhar A, Hind M, Hoffman SC, et al. One explanation does not fit all: a toolkit and taxonomy of AI explainability techniques [Internet]. arXiv. 2019 [cited 2026 Sep 8]. Available from: <https://arxiv.org/abs/1909.03012>

Alvarez-Melis D, Jaakkola TS. Towards robust interpretability with self-explaining neural networks [Internet]. arXiv. 2018 [cited 2026 Sep 8]. Available from: <https://arxiv.org/abs/1806.07538>

Geurts P, Ernst D, Wehenkel L. Extremely randomized trees. *Mach Learn*. 2006;63:3-42.
<https://doi.org/10.1007/s10994-006-6226-1>.

Safavian SR, Landgrebe D. A survey of decision tree classifier methodology. *IEEE Trans Syst Man Cybern.* 1991;21:660-74.
<https://doi.org/10.1109/21.97458>.

Street WN, Wolberg WH, Mangasarian OL. Nuclear feature extraction for breast tumor diagnosis. In: *Proceedings of the Biomedical Image Processing and Biomedical Visualization*; 1993; San Jose, CA. Bellingham (WA): SPIE; 1993. 1905:861-70.
<https://doi.org/10.1117/12.148698>.

Wei D, Dash S, Gao T, Gunluk O. Generalized linear rule models. In: Chaudhuri K, Salakhutdinov R, editors. *Proceedings of the 36th International Conference on Machine Learning*; 2019; Long Beach, CA. *Proceedings of Machine Learning Research*. 2019;97:6687-96.

Chen C, Lin K, Rudin C, Shaposhnik Y, Wang S, Wang T. An interpretable model with globally consistent explanations for credit risk [Internet]. *arXiv*. 2018 [cited 2026 Sep 8]. Available from: <https://arxiv.org/abs/1811.12615>

Alcalá-Fdez J, Fernández A, Luengo J, Derrac J, García S, Sánchez L, Herrera F. KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. *J Mult-Valued Log Soft Comput.* 2011;17(2-3):255-87.