



UNIVERSITÀ
DEGLI STUDI
DI PADOVA

Head Office: Università' degli Studi di Padova
Department of Information Engineering

Ph.D. Course in: Information Engineering
Curriculum: Bioengineering
38th cycle

Trustworthy AI: Exploring Reliability, Transparency and Adaptability of Machine Learning Systems.

Coordinator: Prof. Fabio Vandin
Supervisor: Prof. Gian Antonio Susto
Co-Supervisor: Prof. Simone Del Favero

Ph.D. Student: Davide Sartor

Abstract

Machine learning is becoming deeply embedded in everyday decision making, from digital assistants to safety-critical infrastructure. As these systems grow more pervasive, trust becomes essential. Trustworthy machine learning rests on three pillars: reliability, adaptability, and transparency. Reliability ensures that models behave predictably under distribution shift, resist spurious correlations, and remain resilient to rare events. Adaptability enables systems to adjust to evolving environments, scarce or shifting data, and the feedback loops that naturally arise in real-world deployment. Transparency provides interpretable reasoning and allows for meaningful human oversight. Together, these qualities move trust beyond accuracy alone and toward machine learning systems that are stable, auditable, and aligned with human goals and constraints, even under uncertainty and change. This thesis treats human interaction, control considerations, and structural inductive biases as primary design principles, and is organized into three main parts. First, we incorporate human feedback into anomaly detection models, enabling them to adapt to evolving definitions of abnormality and remain robust against misalignment with expert intent. Second, we investigate the use of safe and interpretable reinforcement learning for safety-critical scenarios, where robustness and reliability of control policies are paramount. Finally, we explore how to encode desirable properties directly into neural network architectures, providing guarantees that improve robustness, transparency, and trustworthiness. Together, these contributions advance the design of machine learning systems that go beyond accuracy, going towards Machine Learning systems that can be safely and confidently deployed in real-world decision making.

Sommario

Il machine learning sta diventando sempre piu' parte integrante dei processi decisionali quotidiani, dai sistemi digitali di assistenza fino alle infrastrutture critiche per la sicurezza. Con la crescente diffusione di questi sistemi, la fiducia diventa essenziale. Questo fondamento si basa su tre pilastri: affidabilita', adattabilita' e trasparenza. L'affidabilita' garantisce che i modelli si comportino in modo prevedibile in presenza di cambiamenti di distribuzione, resistano a correlazioni spurie e rimangano resilienti a eventi rari. L'adattabilita' consente ai sistemi di adeguarsi ad ambienti in evoluzione, a dati scarsi o mutevoli e ai cicli di retroazione che emergono naturalmente nelle applicazioni reali. La trasparenza offre capacita' di ragionamento interpretabile e permette una supervisione umana significativa. Insieme, queste qualita' spostano il concetto di fiducia oltre la sola accuratezza, verso sistemi di machine learning stabili, verificabili e allineati agli obiettivi e ai vincoli umani, anche in condizioni di incertezza e cambiamento. Questa tesi considera l'interazione umana, le esigenze di controllo e i bias induttivi strutturali come principi fondamentali di progettazione, ed articolata in tre parti principali. In primo luogo, incorporiamo il feedback umano nei modelli di rilevamento delle anomalie, consentendo loro di adattarsi a definizioni di anomalia in evoluzione e di rimanere robusti contro disallineamenti rispetto all'intento degli esperti. In secondo luogo, esploriamo l'uso di tecniche di reinforcement learning sicure e interpretabili per scenari critici per la sicurezza, in cui la robustezza e l'affidabilita' delle politiche di controllo sono fondamentali. Infine, indaghiamo come incorporare proprieta' desiderabili direttamente nelle architetture di reti neurali, fornendo garanzie che migliorano robustezza, trasparenza e affidabilita'. Nel complesso, questi contributi fanno progredire la progettazione di sistemi di machine learning che vanno oltre la sola accuratezza, avvicinandosi a sistemi che possono essere utilizzati in modo sicuro e con fiducia nelle decisioni del mondo reale.

List of Publications

- [1] **D. Sartor**, T. Barbariol, and G. A. Susto, Bayesian active learning isolation forest (B-ALIF): A weakly supervised strategy for anomaly detection Engineering Applications of Artificial Intelligence Apr. 2024.
- [2] E. Marcelli, T. Barbariol, **D. Sartor**, and G. A. Susto, Active Learning-based Isolation Forest (ALIF): Enhancing anomaly detection with expert feedback Information Sciences Sep. 2024.
- [3] T. Barbariol, J. Lesouple, **D. Sartor**, J. Y. Tournieret, and G. A. Susto, Optimal Transport for Domain Adaptation in Anomaly Detection Systems submitted to: Engineering Applications of Artificial Intelligence
- [4] V. Zaccaria*, **D. Sartor***, and G. A. Susto, Extended B-ALIF: Improving Anomaly Detection with Human Feedback in 2024 32nd Mediterranean Conference on Control and Automation (MED), Jun. 2024.
- [5] V. Zaccaria*, **D. Sartor***, S. Del Favero and G. A. Susto, "Fault Identification Enhancement with Reinforcement Learning (FIERL)" 2024 American Control Conference (ACC), Toronto, Canada, 2024.
- [6] **D. Sartor***, A. Sinigaglia*, and G. A. Susto. "Advancing Constrained Monotonic Neural Networks: Achieving Universal Approximation Beyond Bounded Activations." Forty-second International Conference on Machine Learning, Vancouver, Canada, 2025.
- [7] A. Arcudi*, **D. Sartor***, A. Sinigaglia, V. Francois-Lavet, G. A. Susto. "Multi-layer Abstraction for Nested Generation of Options (MANGO) in Hierarchical Reinforcement Learning." 7th IFAC Conference on Intelligent Control and Automation Sciences, Padua, Italy, 2025.

- [8] T. Kalb*, **D. Sartor***, F. B. Radwan, V. Zaccaria, C. Masiero, G. A. Susto. "Batch-querying Strategies in Bayesian Active Learning Isolation Forests." 7th IFAC Conference on Intelligent Control and Automation Sciences, Padua, Italy, 2025.
- [9] A. Sinigaglia*, **D. Sartor***, M. Ceccon, G. A. Susto. "Simple and Effective Specialized Representations for Fair Classifiers." Thirty-Ninth Annual Conference on Neural Information Processing Systems (NeurIPS), San Diego, USA, 2025.

* Equal contribution

Contents

1	Introduction	1
I	Anomaly Detection with Human Feedback	3
2	Introduction	4
2.1	Definition of Anomaly	6
2.2	Decision Support Systems	9
2.3	Active learning	10
2.4	Isolation Forest	12
2.5	Contributions	15
3	Active Learning Isolation Forest	17
3.1	Introduction	17
3.2	Related works	18
3.3	Active Learning Isolation Forest	21
3.3.1	Unsupervised Fit	23
3.3.2	Update Step	23
3.3.3	Query Step	25
3.4	Model complexity and overhead	28
3.5	Experimental Results	29
3.6	Conclusions	33
4	Bayesian Active Learning Isolation Forest	38
4.1	Introduction	38
4.2	Bayesian Inference	39
4.3	Bayesian Active Learning Isolation Forest	41
4.3.1	Generalized abstract properties	42
4.3.2	Node property updates	42

4.3.3	Initialization and IF prior	44
4.3.4	Anomaly score	45
4.3.5	Connections with ALIF	47
4.3.6	Query strategy	48
4.3.7	Model complexity and overhead	49
4.4	Results	50
4.4.1	Experiments on synthetic dataset	50
4.4.2	Experiments on real world data sets	54
4.5	Conclusion	57
5	Extended Bayesian Active Learning Isolation Forest	62
5.1	Introduction	62
5.2	Extended Isolation Forest	63
5.3	BALEIF	64
5.4	Time complexity	66
5.5	Experimental Results	66
5.5.1	Synthetic datasets	67
5.5.2	Real-world dataset	69
5.6	Conclusion	70
6	Batch Bayesian Active Learning Isolation Forest	72
6.1	Introduction	72
6.2	Preliminaries	73
6.3	BALD	74
6.4	Closed-form BALD in BALIF	75
6.5	BatchBALD	76
6.6	Closed-form BatchBALD in BALIF	77
6.7	Robust-Worst-Case Batch Querying	78
6.8	Experiments	79
6.9	Conclusion	82
II	Safe and Interpretable Reinforcement Learning	83
7	Fault Identification with Safe Reinforcement Learning	84
7.1	Introduction	84
7.2	System Modelling and Notation	86
7.3	Constrained Reinforcement Learning	87

7.3.1	Constrained Policy Optimization	89
7.4	Fault Identification Enhancement with Reinforcement Learning	91
7.5	Case Study	93
7.5.1	Passive Component Derivation	94
7.5.2	Framework Application	95
7.5.3	Experimental results	96
7.6	Conclusions	99
8	Hierarchical Abstractions for Generation of Options	100
8.1	Introduction	100
8.2	Options in the Literature	102
8.3	Options as Actions in Abstract Graph	103
8.4	Structured Options using Hierarchical Abstractions	104
8.5	Task actions	106
8.6	Learning Process	107
8.7	Experimental Results	109
8.8	Conclusion	112
III	Encoding Desirable Properties into Models	114
9	Fair Representation Learning	115
9.1	Introduction	115
9.2	Related Work	119
9.2.1	Variational Autoencoders	119
9.2.2	Adversarial Training	119
9.2.3	Other Approaches	120
9.3	Background	121
9.3.1	Statistical Distance and Adversarial Evaluation	121
9.3.2	Additional Fairness Metrics	122
9.3.3	Logistic Regression	125
9.4	Fair Representations matching Characteristic Function	125
9.4.1	Characteristic Function Distance	125
9.4.2	Comparison to Adversarial Learning and FNF	127
9.5	Fair Classification matching Sufficient Statistics	127
9.5.1	Connection to Distribution Moments	128
9.5.2	Simplified Penalty for Classification	129
9.5.3	Fairness Guarantees	133

9.6	Connections to Maximum Mean Discrepancy	134
9.7	Matching All Moments	135
9.8	Experimental Evaluation	135
9.8.1	Visualization of Learned Representations	138
9.8.2	Comparison on Adult Dataset	138
9.8.3	Comparison on German Dataset	140
9.8.4	Comparison with FNF	141
9.8.5	Evaluation of Fair Generative Models	143
9.9	Conclusions	143
10	Monotonic Neural Networks	145
10.1	Introduction	145
10.2	Related work	149
10.2.1	Hard Monotonicity	149
10.2.2	Soft Monotonicity	150
10.3	Monotone MLP	150
10.4	Known conditions for universal approximation	151
10.5	Naive bound for universal approximation of alternating MLPs	152
10.6	Universal approximation theorem for non-threshold activations	152
10.7	Non-positive constrained monotonic MLP	158
10.8	Relaxing weight constraints	160
10.9	Extension to other activations	163
10.10	Experimental results	165
10.10.1	Comparison of different parametrizations on toy datasets	165
10.10.2	Comparison on Real-world Datasets	168
10.11	Initialization Vanishing Gradient in Constrained MLPs	172
10.12	Conclusions	174
11	Conclusions and Discussion	175
11.0.1	Anomaly Detection with Human Feedback	176
11.0.2	Safe and Interpretable Reinforcement Learning	176
11.0.3	Encoding Desirable Properties into Models	177
11.0.4	Final Discussion	178

1

Introduction

Machine learning (ML) has rapidly evolved into one of the most powerful drivers of technological innovation, shaping domains as diverse as health-care, finance, transportation, and industrial automation. As these systems increasingly influence decisions with direct societal and economic impact, the question of trust has become central. Accuracy alone is no longer sufficient to guarantee the safe and responsible deployment of ML models. Instead, robustness, transparency, and adaptability have emerged as essential pillars for what is now referred to as trustworthy machine learning.

Trust is vital because ML systems are now embedded in contexts where errors can have profound consequences. In healthcare, predictive models are used to assist with disease diagnosis or treatment planning; if a model is biased or poorly calibrated, the resulting misdiagnosis may directly affect patient outcomes. In finance, credit scoring and fraud detection algorithms decide who gains access to capital and who is flagged as a risk, raising concerns of fairness and accountability when decisions systematically disadvantage particular groups. In autonomous driving and industrial control, robustness is paramount: a misclassification of sensor data or an unsafe control policy can lead to accidents with potentially catastrophic results. Even in everyday applications, such as recommendation systems and digital assistants, lack of

transparency can erode user trust and amplify societal biases by reinforcing stereotypes or spreading misinformation.

Robustness ensures that systems remain reliable even under distributional shifts, adversarial manipulation, or rare and unforeseen events. Transparency addresses the demand for interpretability, enabling human operators to understand, audit, and challenge machine reasoning. Adaptability acknowledges the dynamic nature of real-world environments, where models must learn and adjust to new data, shifting conditions, and continuous human feedback. Together, these principles form the foundation for ML systems that are not only accurate, but also fair, accountable, and aligned with human values and regulatory requirements such as the GDPR [EC] and the EU AI Act [Cou14]. This thesis is motivated by the need to design ML methods that embody these qualities. To this end, it advances trustworthy machine learning along three complementary dimensions:

- **Anomaly Detection with Human Feedback:** We investigate interactive approaches to anomaly detection that incorporate expert knowledge, enabling models to adapt to evolving definitions of anomalies and improve robustness to misalignment.
- **Safe and Interpretable Reinforcement Learning:** We develop frameworks for decision-making under uncertainty that respect safety constraints while remaining interpretable to human operators, with a particular focus on fault diagnosis in safety-critical systems.
- **Encoding Desirable Properties into Models:** We propose architectures and representation learning methods that embed fairness and monotonicity guarantees, offering theoretical robustness while maintaining practical performance and stability.

By addressing reliability, transparency, and adaptability across these three perspectives, this work contributes both theoretical insights and practical algorithms for building machine learning systems that can be safely and confidently deployed in real-world decision-making.

Part I

Anomaly Detection with Human Feedback

2

Introduction

Machine Learning (ML) has achieved remarkable success over the past decades, becoming one of the most prominent and widely studied branches of artificial intelligence. Its methods are now deeply embedded across diverse research areas and real-world applications, where the primary objective is to derive predictive functions from input data in order to make accurate forecasts on previously unseen cases. In particular, ML has transformed numerous industries, driving innovation in fields such as Industry 4.0 [AMN⁺19], healthcare [MKMM20], and transportation [LB⁺23].

Within this broad landscape, Anomaly Detection (AD), also known as outlier detection, stands out as a critical subfield of ML. AD focuses on identifying patterns or data points that significantly deviate from the expected normal behavior observed in the majority of data [Haw80, CBK09]. These deviations often correspond to rare but impactful events, making their timely detection essential. By signaling irregular or unexpected behavior, AD plays a fundamental role in monitoring complex systems, predicting failures, and preventing costly disruptions. This is especially relevant in industrial contexts, where data-driven approaches such as predictive maintenance rely on AD to isolate faults and reduce the risk of sudden breakdowns. Importantly, the model-free nature of many AD approaches, driven by empirical data

rather than explicit system modeling, makes them particularly suitable for dynamic, high-dimensional, or poorly understood processes.

The versatility of AD explains its widespread adoption across domains. In cybersecurity, anomalies may indicate intrusions or network breaches [YLL⁺22]; in industrial systems, they can reveal sensor irregularities that signal equipment malfunctions [CMF20]; and in healthcare, they help detect unusual physiological patterns such as abnormal heart rates [SKHN21]. Beyond these cases, AD has become indispensable for credit card fraud detection [CLC⁺21], traffic monitoring [JGLM23], medical diagnostics [FGD⁺21], and intrusion detection in computer networks [NTND21]. Its breadth of applications, from fault detection in automated processes [CMBS19] to safeguarding financial transactions, demonstrates both its robustness and its central role in modern intelligent systems.

As in other areas of machine learning, approaches to anomaly detection can be broadly categorized into supervised, semi-supervised, and unsupervised methods. Supervised approaches require large amounts of labeled data to train reliable models. However, this is rarely feasible in practice: while normal operating data may be abundant, accurate labels for the rare anomalous class are scarce, costly, and time-consuming to obtain. Labeling is often performed by domain experts through labor-intensive manual inspection, and even then, anomalies are inherently difficult to define and detect. As a result, supervised methods are relatively uncommon in anomaly detection. Semi-supervised approaches alleviate this issue by leveraging partially labeled data, typically assuming that labels are available for normal instances while anomalies remain unlabeled. Although this setting reduces the labeling effort, it still requires some degree of expert annotation, which may not be practical in large-scale or rapidly evolving domains. Consequently, most real-world applications rely on unsupervised anomaly detection, where no ground-truth labels are provided during training [ZG09, ZHBN19]. In this setting, the fundamental assumption is that normal data significantly outnumber anomalies and that anomalies exhibit measurable differences compared to inliers. Unsupervised methods exploit structural properties of the data to distinguish unusual patterns without explicit guidance from labeled examples.

Several families of unsupervised techniques have been proposed. Density-based methods, such as LOF [BKNS00], COF [TCFC02], LoOP [KKSZ09], and LOCI [PKG03], assume anomalies lie in low-density regions of the feature space. Distance-based approaches, including k-NN [KN98], ABOD [KSZ08], and RBRP [GPO08], define anomalies as points far from their neigh-

bors according to some distance metric. Clustering-based methods, such as OFP [JTS01] and CBOD [JA08], identify data points that do not belong to any coherent cluster. A particularly influential family of methods is isolation-based approaches, exemplified by Isolation Forest (IF) [LTZ08, LTZ12]. Unlike most algorithms that attempt to model normal behavior, IF explicitly aims to isolate anomalies by exploiting the assumptions that anomalies are few in number and possess attribute values that differ significantly from normal data. This inversion of perspective has made IF and related methods especially popular in practical anomaly detection tasks.

2.1 Definition of Anomaly

Anomaly detection focuses on identifying data points that exhibit behaviors distinct from the majority of the dataset. Although widely studied, there is still no universally accepted definition of what constitutes an anomaly. A commonly cited description is that an anomaly (or outlier) is an observation that deviates so much from other observations as to raise suspicion that it was generated by a different mechanism [Haw80]. More recent perspectives emphasize that the definition of an anomaly often varies by task and application [Foo21].

The motivations for identifying anomalies are diverse. In some cases, anomalies are considered noise and are removed to improve the robustness of predictive models or statistical measures. In others, such as fault detection or monitoring applications, anomalies are precisely the points of interest, as they may indicate rare or critical events [AAO17, KEE⁺15, ZHHL20, CHA⁺23]. This dual perspective highlights why anomalies cannot be defined in absolute terms but must instead be understood relative to the goals of the analysis. Recent literature has also drawn a distinction between outliers and anomalies [DWD⁺16, SSK21]. Outliers are points identified by unsupervised models as statistically deviant from the bulk of the data distribution. Anomalies, on the other hand, are those data points judged by users or domain experts to be truly abnormal in context. While related, the two do not always overlap: outliers detected automatically may not align with what practitioners consider meaningful anomalies, underscoring the importance of domain knowledge in interpreting results.

Indeed, anomalies are highly domain-specific [Foo21, SSK21]. The same dataset may yield different anomalies depending on the application: a manufacturer might define anomalies as instrument malfunctions, whereas an

end-user may view anomalies as unexpected variations in the monitored process [BFS20]. This context dependence complicates both the definition and detection of anomalies, often requiring labeled data that reflect the specific users notion of abnormality. As a consequence, designing domain-specific anomaly detectors is a nontrivial task that depends heavily on expert input and evolving operational requirements [ZVF⁺23]. Figure 2.1 illustrates this challenge with a two-dimensional projection of the vowels dataset [She16a]. Here, anomalies are not simply isolated points in sparse regions of the space, but rather form a coherent class defined by domain experts. Without user feedback, an automatic detector relying solely on statistical deviation would likely miss such patterns. This example demonstrates the critical interplay between data structure, domain expertise, and application context in defining and detecting anomalies.

While unsupervised anomaly detection methods are widely used, relying on them alone can be risky and often misleading. As discussed earlier, anomalies are highly domain-dependent, meaning that an unsupervised detector may fail to identify anomalous data points that are relevant in context, or conversely, may flag normal points as anomalous [DWD⁺16, SSK21]. This mismatch arises because unsupervised algorithms adopt generic, model-driven definitions of anomalies, which do not necessarily align with the end-users expectations or domain-specific needs. Each unsupervised approach embodies a particular inductive bias, implicitly defining what constitutes an anomaly based on its detection strategy [HHH⁺22a]. For instance, density-based models such as Local Outlier Factor (LOF) identify anomalies as points lying in local low-density regions [BKNS00], while angle-based methods like ABOD flag points with low variance in angular relationships [KSZ08]. Isolation Forest (IF), one of the most popular shallow AD models, defines anomalies as data points that are easier to isolate compared to the majority [LTZ08].

However, such assumptions may not hold in practice: anomalies are not always sparse or isolated, and in some contexts they may form dense clusters, rendering the IF bias ineffective. Similar limitations affect distance-based and probabilistic methods, whose success depends heavily on whether their implicit definitions of anomalies match the actual data characteristics. This misalignment between algorithmic bias and user expectations can lead to poor accuracy and reduced interpretability [SBS24a]. As emphasized in [SSK21], incorporating domain knowledge is essential for selecting appropriate models and encoding user-specific definitions of anomalies. While supervised approaches could directly capture such definitions, their reliance

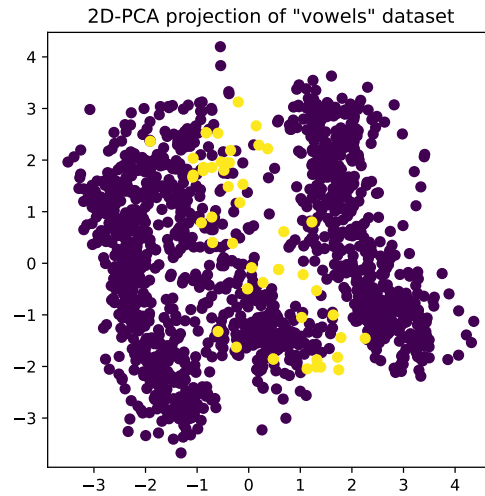


Figure 2.1: Projection of the *vowels* dataset onto two dimensions using Principal Component Analysis. Purple points represent normal data, while yellow points indicate anomalies. Two aspects emerge: (i) anomalies cannot be separated from normal data when considering only two features, and (ii) anomalies tend to form a specific class distinct from general outliers. In this example, not all points in low-density regions or far from the majority are anomalous, only those located within a particular region of the space, as defined by domain expertise. Moreover, anomalies do not necessarily exhibit obvious discrepancies from normal points, often overlapping with them in feature space. As a result, detecting anomalies in this scenario poses a significant challenge for unsupervised methods.

on labeled datasets makes them impractical in many real-world applications. As a result, unsupervised methods like IF remain popular due to their simplicity and scalability, but they often struggle to reflect user-driven anomaly definitions.

2.2 Decision Support Systems

Decision Support Systems (DSS) [SGC17] are designed to analyze continuous streams of data and support users in making rapid, informed decisions on complex problems. In these systems, human operators play a central role: they monitor the process, interact with the system, and ultimately act as the decision makers [CMS⁺02, LG04]. Because expert users are already engaged in overseeing the data, their feedback can be leveraged not only to enhance the informativeness of the DSS, but also to improve the underlying anomaly detection models.

We adopt two assumptions in this context: (i) users can provide labels to the model, and (ii) these labels are accurate. Assumption (ii) is reasonable in many settings, medical feedback can be provided by trained physicians, while annotations in industrial processes can be made by equipment experts. Future extensions may relax this assumption to account for noisy or imperfect labels. Under these conditions, models that can query experts and adapt based on their feedback become highly appealing. A natural way to operationalize this interaction is through Active Learning (AL), a paradigm in which the model actively queries a human teacher or oracle (in this case, the domain expert) for labels on selected data points [PM04]. Instead of requiring full annotation of the dataset, AL focuses on the most informative instances, thereby reducing labeling costs while quickly improving model performance. This is particularly advantageous in DSS environments, where analysts already monitor real-time data streams [DMC⁺23].

By querying experts for labels on a limited number of strategically chosen data points, anomaly detection models can incrementally refine their accuracy, aligning their outputs with user expectations. Such human-in-the-loop learning perspectives make AL a well-suited paradigm for DSS. Although AL has been extensively studied in classification tasks [DSM09, LYQ⁺23], its application to anomaly detection is more recent, with promising results [DWD⁺16]. In parallel, semi-supervised approaches have been proposed to bridge the gap between supervised and unsupervised anomaly detection. In many practical cases, only a small number of annotations can be acquired,

insufficient for supervised training but still valuable for guiding unsupervised algorithms toward user-relevant anomalies. This scenario is typically framed as weakly supervised (WS) anomaly detection. WS methods exploit limited or coarse labels to refine model predictions, for example by aligning them with user-defined notions of abnormality. Recent work in this direction has explored deep anomaly detection frameworks that incorporate pseudo-label generation [RVG⁺19], as well as domain-specific strategies, such as anomaly detection in video streams [ZLQ⁺23]. Together, AL and WS approaches demonstrate how integrating expert feedback, even in limited quantities, can substantially improve the alignment between anomaly detection models and domain-specific requirements.

2.3 Active learning

In many fields, the application of machine learning (ML) techniques is severely limited by difficulties in the labeling process [ZG09, GY19]. Large-scale unlabeled datasets may be available, but obtaining accurate labels is often costly, requiring expert intervention, or in some cases impossible. Anomaly detection (AD) is a prime example: anomalies are inherently rare and difficult to identify, making accurate labeling a challenging task. For this reason, most approaches rely on unsupervised algorithms, where labels are not required.

Active Learning (AL) is a subset of ML in which models interact with an oracle capable of providing labels for specific data points. There are many variants of AL setups, but in its simplest formulation, a model has access to a large unlabeled dataset and can subsequently query the oracle for the correct labels of selected points [DSM09]. In practical terms, this can represent a user providing targeted feedback to improve model performance with minimal labeling effort. The key idea is that the model can direct labeling efforts strategically, thereby minimizing the costs associated with expensive annotation processes.

To illustrate the advantages of AL, consider a simple binary classification problem with linearly separable points uniformly distributed in the unit interval. Suppose a large unlabeled dataset is available, but only k labels can be obtained due to a limited labeling budget. The goal is to identify the decision boundary between the two classes. Training a Support Vector Machine (SVM) on k randomly labeled points will produce a decision boundary approximately halfway between the most extreme labeled points from each class. Analytically, the expected distance of the closest labeled example of

each class to the true boundary is proportional to $1/k$, meaning the model learns slowly as k increases. The main reason is that new labels outside the uncertainty region provide little information, and as the error decreases, the probability of randomly sampling an informative point diminishes.

Active Learning addresses this inefficiency by selectively querying points near the midpoint of the uncertain region. Each new label halves the maximum error, leading to an exponential decrease in error proportional to 2^{-k} . Essentially, AL performs a binary search for the optimal decision boundary. While this idealized one-dimensional example does not directly extend to high-dimensional or highly unbalanced anomaly detection datasets, the underlying intuition remains relevant and informs the design of practical AL strategies [LBST21b]. It is important to note that identifying the optimal points for querying can be computationally expensive, requiring evaluation of model outputs across the entire dataset or solving complex optimization problems. In practice, heuristic strategies are often used to reduce this overhead. In later Sections, we rely on such heuristics, focusing on practical, computationally efficient approaches rather than exact error bounds.

Many ML tasks, including anomaly detection, suffer from label scarcity. Anomalies are rare, producing highly imbalanced datasets dominated by normal instances. Labeling anomalies often requires domain expertise and careful inspection, making the process both time-consuming and costly. In this context, AL emerges as a class of techniques where the algorithm actively queries an information source, typically a human, for labels of representative data points [DSM09]. The primary objective is to maximize model performance while minimizing the number of queries or adhering to a pre-defined labeling budget. Query strategies in AL are diverse and typically based on practical heuristics [KG20]. Although theoretically optimal queries can be defined, such approaches are usually computationally expensive. A common heuristic is to query points for which the model prediction is most uncertain. In ensemble models like Isolation Forest (IF), uncertainty can be assessed by the variability of individual predictors; however, this has been shown to perform suboptimally in practice [MBS22]. In anomaly detection, datasets are highly unbalanced, making a more effective heuristic to query the points with the highest predicted anomaly scores. Since anomalies are rare, these points are more informative than normal points, a strategy employed by methods such as AAD. Nevertheless, this approach may fail if the initial model predictions are unreliable. A more robust alternative is to select instances near the decision boundary, optionally weighted by prediction uncertainty [SBS24a].

In this work, following [SBS24a], we adopt this latter strategy to balance informativeness and robustness while keeping computational costs manageable.

2.4 Isolation Forest

This study is primarily focused on the Isolation Forest (IF) [LTZ09, LTZ08] model. Isolation Forest is an unsupervised ensemble algorithm consisting of multiple decision trees, each designed to isolate anomalies through a process of random hierarchical partitioning of the feature space. In each tree, the data is recursively split along randomly selected features until either a single point is isolated or a predefined maximum depth is reached. The core principle underlying IF is that anomalies, being rare and possessing distinctive attribute values, generally require fewer splits to become isolated and therefore tend to reside in leaves closer to the root.

Isolation Forest has become one of the most widely adopted unsupervised anomaly detection (AD) methods, owing to several advantages: computational efficiency, low memory requirements, and robust performance with minimal hyperparameter tuning [HHH⁺22a]. Its popularity also stems from its conceptual simplicity and scalability, making it particularly suitable for high-dimensional datasets and resource-constrained environments. Introduced by [LTZ08], IF represented a novel paradigm in anomaly detection, departing from traditional methods that primarily rely on distance metrics [Edw98], density estimation [BKNS00], or parametric data modeling [HM10]. Instead of constructing a model of normal data and labeling deviations as anomalies, Isolation Forest leverages the concept of isolation, turning the natural scarcity of anomalies into an advantage. By focusing directly on anomalies rather than on nominal behavior, IF identifies points that are statistically easier to separate from the bulk of the data. This non-parametric, ensemble-based approach offers a cost-effective yet highly effective solution for unsupervised AD [LTZ08, LTZ12]. Isolation Forest has consistently demonstrated strong performance across a variety of real-world applications, often surpassing more complex state-of-the-art methods [SBM17]. Even in modern settings, IF remains one of the top-performing unsupervised anomaly detection techniques, combining speed, scalability, and accuracy while requiring no prior assumptions about the underlying data distribution [HHH⁺22b].

Similarly to Random Forest [Ho95], the Isolation Forest (IF) constructs an ensemble of binary decision trees, known as isolation trees (iTrees). Un-

like Random Forest, this process is fully unsupervised, leveraging the fact that anomalies, due to their distinctive characteristics, are easier to isolate through random partitions. The core concept underlying IF is simple yet effective: anomalies are generally rare and exhibit feature values that are substantially different from normal instances. Consequently, they are likely to be isolated quickly.

During training, the algorithm recursively partitions the feature space using random binary splits until each data point is isolated in a distinct region. These partitions are efficiently represented using binary trees (iTrees), and multiple iTrees are combined to form the iForest, an ensemble of n_T independent trees generated from multiple sub-samples of the dataset. Given an unlabelled dataset $X \subset \mathbb{R}^d$, each iTree is built from a random sub-sample $X' \subset X$ of size ψ . This bagging procedure reduces overfitting and speeds up computation.

The construction of each iTree is as follows: The subset of data X' serves as the root of the tree. Then, one feature is chosen at random and a corresponding split value uniformly within the observed range. The data is then splitted into two subsets based on the selected threshold: points with feature values less than or equal to the threshold go to the left child, and the rest to the right. The last two steps are repeated for each child node in a recursive partitioning process that continues independently for each child node using randomly selected features and split values, stopping when either a single data point occupies a node or a maximum depth h_{max} is reached.

Terminal nodes, or leaves, correspond to distinct regions of the data space. Anomalies, being "few and different," typically require fewer splits to isolate, resulting in shallower terminal nodes compared to normal points. Each node in a tree corresponds to a specific region of the feature space, and the collection of all leaf nodes $L_1, L_2, \dots, L_m$ forms a partition of the space. For each data point x , there exists a unique path from the root to a single leaf node $L_{x,T}$ in each tree T of the ensemble. If the context allows, the subscript notation may be simplified to reduce clutter.

The algorithms complexity is independent of the full dataset size due to the fixed sub-sample size ψ . Also, the process is not affected by the number of features d , as splits are chosen randomly from the feature set. Finally, each leaf node represents a well-defined region of the data space, ensuring that each point belongs to exactly one leaf per tree, enabling straightforward anomaly scoring based on path lengths.

Using the same notation, the depth of a leaf node in a given tree T is

defined as the depth of the corresponding node in the tree:

$$h(L) = \text{depth of } L. \quad (2.1)$$

The maximum height of a tree, h_{max} , is set to $\lceil \log_2 \psi \rceil$, corresponding to the height of a complete binary tree with ψ nodes. This design choice ensures a fast and bounded computational structure, independent of the size of the original dataset. The cost of constructing the entire ensemble of n_T trees is therefore $O(n_T \psi)$, since each tree requires at most ψ operations.

Once the iForest is trained, the anomaly score $a(x)$ of a data point x , representing the likelihood that x is anomalous, is computed based on the average depth of x in the ensemble of trees:

$$a(x) = 2^{\frac{-\mathbb{E}[h(L_x)]}{c(\psi)}}, \quad (2.2)$$

where $h_T(x)$ denotes the depth of the leaf node containing x in tree T , and $c(\psi)$ is a normalization factor defined as

$$c(\psi) = 2H(\psi - 1) - 2\frac{\psi - 1}{\psi}, \quad (2.3)$$

with $H(j)$ representing the j -th harmonic number. This factor corresponds to the expected average path length of an unsuccessful search in a binary search tree of size ψ [LTZ08]. For practical purposes, $H(j)$ can be approximated as $\log(j) + 0.5$.

The interpretation of the anomaly score is intuitive:

1. If $\mathbb{E}_{T \in F}[h_T(x)] \rightarrow \psi - 1$, then $a(x) \rightarrow 0$, and x is likely normal.
2. If $\mathbb{E}_{T \in F}[h_T(x)] \rightarrow 0$, then $a(x) \rightarrow 1$, indicating a high probability that x is anomalous.
3. If $\mathbb{E}_{T \in F}[h_T(x)] \approx c(\psi)$, then $a(x) \approx 0.5$, suggesting the point is neither clearly normal nor anomalous.

The anomaly score can be converted into a binary classification by selecting an appropriate threshold. However, determining an optimal threshold is highly data-dependent and can be challenging in practice [HS18]. In essence, each iTree can be viewed as a weak predictor, providing a score $2^{-h_T(x)/c(\psi)}$ for each point, with the final IF score computed as the geometric mean of

the ensemble predictions. Shorter average path lengths correspond to higher anomaly scores, reflecting the principle that anomalies are easier to isolate than normal points.

This construction ensures that IF is both computationally efficient and robust, scalable to high-dimensional datasets, and independent of the full dataset size due to the fixed sub-sample ψ . At the same time, it provides a principled and interpretable mapping from tree structure to anomaly likelihood, enabling effective identification of outliers in diverse applications [LTZ08].

To improve computational efficiency, the tree-splitting procedure can be truncated when the depth reaches a maximum value

$$h_{max} = \lceil \log_2 \psi \rceil. \quad (2.4)$$

The rationale is that points with depth greater than h_{max} are likely normal, and their true depth can be approximated using a residual depth correction. This residual term represents the expected additional depth that would have been obtained if the tree had continued growing, and it is equal to $c(n)$, where n is the number of data points in the terminal region under consideration. Importantly, this correction accounts for the bias introduced by truncating the tree. For regions where isolation occurs naturally before reaching h_{max} , the residual depth is zero; otherwise, it grows approximately logarithmically with n , reflecting the expected path length of points in a complete binary search tree of size n . With this structure, the computational complexity of evaluating the anomaly score for a single point x is $O(n_T \log_2 \psi)$, corresponding to the cost of traversing the path of x through each of the n_T trees. An additional advantage of this ensemble-based approach is its inherent parallelization capability: both the construction of trees and the computation of anomaly scores can be fully parallelized, since each tree operates independently. This ensures scalability and efficiency even for high-dimensional or large-scale datasets.

2.5 Contributions

In this work, we specifically focus on enhancing the Isolation Forest (IF) algorithm by incorporating expert feedback to better align its predictions with domain-specific definitions of anomalies. Isolation Forest, while efficient and widely used, relies on assumptions about anomaly sparsity and separability that do not always hold in practical contexts. To address this, we investi-

gate mechanisms that allow the model to adapt its isolation process based on user-provided labels, effectively modifying the inductive bias of the algorithm.

Our approach leverages both active learning and weakly supervised strategies to make expert input actionable. Through selective querying of the most informative points, the model can iteratively refine its decision boundaries, while even small amounts of labeled data can be used to guide the scoring mechanism toward anomalies that matter in the specific application domain. By integrating human feedback in this way, we enhance the ability of IF to detect complex or context-dependent anomalies that would otherwise be overlooked or misclassified.

This work demonstrates that classical anomaly detection models like Isolation Forest can be substantially improved by embedding expert knowledge directly into their operation. The proposed enhancements provide a path toward more adaptive, user-aligned anomaly detection, offering both higher accuracy and greater interpretability in domain-specific scenarios. These results highlight the potential of integrating human expertise into the model, establishing a framework for future exploration of expert-guided, human-in-the-loop anomaly detection.

3

Active Learning Isolation Forest

3.1 Introduction

The detection of anomalous behaviors has become an emerging need in many applications, particularly in contexts where security, reliability, and fault prevention are critical. While the definition of an anomaly varies across domains, obtaining a fully labeled dataset is often impractical or prohibitively time-consuming. Unsupervised models, commonly used to overcome the lack of labels, frequently fail to capture domain-specific anomalies because they rely on general definitions of outliers.

To address this limitation, this chapter introduces a novel approach called Active Learning-based Isolation Forest (ALIF). ALIF reduces the number of required labels while tuning the detector to the users domain-specific definition of anomalies. The method is particularly appealing in scenarios where users can interact with the system and provide feedback. Many smart monitoring systems embed unsupervised anomaly detection capabilities but lack mechanisms to adjust predictions based on operational feedback. ALIF enhances such systems by exploiting user input during normal operations, en-

abling the model to adapt incrementally to the anomalies that matter most.

The core idea of ALIF is to iteratively query labels for the most informative or significant data points, minimizing human interaction while maximizing detection performance. This approach frames ALIF as an Active Learning-based model [KG20]. Active Learning is particularly effective when labeled samples are scarce or expensive to obtain. Its key principle is that high-accuracy results can be achieved if the algorithm is allowed to select which points to label and learn from them [Set11]. At each iteration, the model is updated with the newly labeled data and queries the next set of points considered most informative. This process continues until either the model reaches sufficient performance or a labeling budget is exhausted.

ALIF offers two important advantages over conventional supervised and computationally intensive approaches. First, it can begin operating without any labeled data, leveraging an initial unsupervised training phase. Second, it can effectively learn even when labeled instances are available for only one class, which is a key benefit when anomalies are rare or costly to annotate.

This work specifically focuses on Isolation Forest, one of the most effective unsupervised anomaly detection algorithms [HHH⁺22a], and proposes strategies to improve its performance by incorporating expert feedback. In particular, we compare two AL query policies for selecting which points to label and two alternative strategies for updating the internal structure with minimal computational cost. The proposed strategies aim to maximize detection performance while minimizing labeling effort and ensuring efficient model updates, demonstrating that expert-guided improvements can significantly enhance the practical effectiveness of Isolation Forest in real-world, domain-specific applications.

By combining the efficiency of Isolation Forest with targeted expert guidance, ALIF provides a practical and scalable solution for domain-adaptive anomaly detection. This chapter lays the foundation for a more interactive, human-in-the-loop approach to anomaly detection, offering models that are both statistically robust and aligned with domain-specific requirements.

3.2 Related works

In the field of anomaly detection, most approaches rely on unsupervised models due to the difficulty of obtaining human annotations. However, in some cases, a relatively small, and possibly imperfect, labelled dataset may be available. This scenario is referred to as Weakly Supervised Anomaly

Detection (WSAD). The term is used with slightly different meanings depending on the application. For instance, in video anomaly detection, it often refers to identifying frames of abnormal events when only video-level labels are available [ZLQ⁺22, WWL⁺23, CTC⁺22]. In such contexts, models frequently employ self-supervised techniques to generate pseudo-labels. In this chapter, we focus on WSAD for tabular datasets, where the term simply refers to models capable of leveraging a few labelled instances [PSJvdH20].

Recently, research has increasingly explored WSAD methods that incorporate step-by-step expert feedback into unsupervised detection models, such as One-Class SVM (OCSVM) or Isolation Forest (IF). Despite numerous variants of the original IF, most are not designed to handle an actively supervised scenario. While Active Learning (AL) has been widely employed in classification tasks [RXC⁺21], its application to anomaly detection is relatively recent. Although some studies have explored AL in time series data [WO21], this work focuses on AL for tabular anomaly detection. Notably, few existing methods propose mechanisms to incrementally incorporate incoming labels into unsupervised AD models. Such approaches offer the advantage of preserving the beneficial properties and structure of the base models while improving prediction quality through targeted expert feedback.

In [DWD⁺16], the authors propose an interactive and iterative mechanism to adapt the predictions of the anomaly detection model LODA [Pev16] using expert feedback. Typically, AD models assign an anomaly score to each data point, representing its degree of outlyingness, and the top B most anomalous points are presented as alarms to the analyst. However, due to model-user mismatch, a significant portion of these B alarms can be false positives. To address this issue, the authors introduce Active Anomaly Discovery (AAD), a method that leverages user feedback to iteratively maximize the number of true anomalies among the top B alarms. LODA, being an ensemble of histograms, computes the final anomaly score as the average across individual histogram scores. AAD modifies this combination by introducing weighted averaging, where the weights are optimized iteratively using expert feedback, with the goal of increasing the scores of labeled anomalies relative to labeled normal points.

This concept was later extended to Isolation Forest in [DWF⁺17], resulting in IF-AAD. In this method, rather than adjusting histogram weights, weights are associated with the leaves of the trees. The fundamental tree structure of IF remains unchanged; only the leaf weights are updated. Points with the highest anomaly scores are presented to an expert analyst iteratively,

and the weights are updated via an optimization procedure to ensure that labeled anomalies receive higher scores than labeled normal points. Importantly, the hierarchical partitions of the original Isolation Forest are preserved, and only the weights are modified. This approach allows the model to incorporate expert knowledge incrementally without altering its unsupervised base structure, effectively bridging unsupervised detection with weak supervision through active feedback.

A fundamentally different approach, not based on tree ensembles but on Support Vector Machines (SVMs), has also been explored. In particular, [LBST21b] proposes a semi-supervised anomaly detection algorithm, called ν -SSVM, which aims to improve the unsupervised model provided by a One-Class SVM (OCSVM) [SWS⁺99] using expert feedback in the form of labeled data points. The ν -SSVM framework incorporates labeled instances by modifying the standard OCSVM optimization problem to compute the support vector coefficients. If no labeled data are available, the formulation reduces to the original OCSVM. Conversely, if all points are labeled, it behaves like a standard supervised SVM while accounting for the class imbalance inherent in anomaly detection problems.

This method explicitly balances between unsupervised and supervised approaches, allowing it to leverage limited labels effectively while retaining robustness when labels are scarce. Two hyperparameters govern the number of support vectors and the allowed misclassification of labeled examples, for which default values are suggested by the authors. This semi-supervised extension thus provides a flexible mechanism to incorporate expert feedback incrementally, improving anomaly detection performance without requiring full supervision.

In the Decision Support Systems (DSSs) perspective, solving an optimization problem every time new labels arrive can be unfeasible. This motivation is the core of Active Learning Isolation Forest (ALIF) [MBS22]. The authors propose an efficient method to incrementally update Isolation Forest (IF) as new labels are introduced, under the Active Learning (AL) setting. They refine the anomaly score of IF partitions of the domain space using heuristics, exploiting the proportion of labeled anomalies within each region relative to the total labeled point. ALIF introduces minimal overhead required to update IF but with overall better performances.

Model	Based on	Description
AAD [DWF ⁺ 17]	IF	Assigns weights to each leaf of the IF. Leaf weights determine the model predictions. Computes weights by solving an optimization problem, so that labelled data is separated.
ν -SSVM [LBST21b]	OCSVM	Provides a trade-off between OCSVM and ν -SVM. Computes SV coefficients using a modified ν -SVM objective function (quadratic optimization problem). Scales poorly with dataset size.
ALIF [MBS22]	IF	Updates IF anomaly scores incrementally under the Active Learning setting. Refines partition scores heuristically using proportions of labelled anomalies, avoiding costly optimization and enabling efficient updates suitable for DSSs.

3.3 Active Learning Isolation Forest

As it was explained in previous sections, the field of anomaly detection can largely benefit from incorporating active learning techniques when feasible. For this reason we would like to develop models that can work well in a mostly unsupervised manner, but that are able to incorporate a few labelled data seamlessly and efficiently. In this chapter we present a novel approach, called Active Learning-based Isolation Forest (ALIF). Similarly to IF-ADD, our model is equivalent to a standard Isolation Forest, when no labels are provided, but is able to incorporate user feedback into the model predictions.

However, ALIF differs from the aforementioned works because it is specifically designed to be inserted into an active learning loop, relying on local model updates. Unlike IF-AAD, Random Forest and especially ν -SSVM, the proposed approach does not require the solution of an optimization problem, significantly simplifying the update procedure. Active Learning Isolation For-

est (ALIF) [MBS22], is an active learning algorithm that aims at achieving such goal by exploiting a few simple, yet clever ideas. The proposed approach relies on extending an Isolation Forest model, allowing the possibility to query domain experts for labels.

As the name suggests, ALIF is an extension of the isolation forest algorithm, focusing on allowing the model to adjust to labelled data in a sequential manner. This approach differs from others for its simplicity. In fact, the update procedure is very light and does not require the solution of complex optimization problems, but instead relies on simple heuristic rules.

In order to make this possible, ALIF starts from a standard isolation forest with n_T trees obtained from sub samples of size ψ as base model, but at each node, it also stores few additional information related to labelled data. In the original paper [MBS22], the authors only define the "color" of a node, but for the sake of this discussion, and to ease the extensions that will be presented later in this Section, it will be useful to generalize the idea, allowing for any kind of information to be stored as abstract node proprieties.

This extra information is then used to adjust the anomaly score provided by the IF, obtaining the ALIF model predictions. Under this framework, once the base Isolation Forest is fully grown, the algorithm is able to select a data point that should be labelled. The selected point is then presented to a human operator who manually inspects and classifies it. Based on the result of this querying process, the model structure is then modified adjusting its predictions. In order for this process to be efficient, at every iteration the main structure of the Isolation Forest remains untouched, and only the anomaly score calculation is modified.

In the following we will analyse each step of ALIF, which can be broken down into three interconnected, yet ultimately independent, tasks:

- i. *Unsupervised Fit*: Fitting the baseline Isolation Forest model given an unlabelled data set.
- ii. *Update Step*: The process by which a newly acquired label is ultimately used to modify the model prediction;
- iii. *Query Step*: The process that determines which point is presented to the domain expert for labelling.

3.3.1 Unsupervised Fit

As discussed earlier, the central idea of ALIF is to integrate labeled data into the model by modifying the abstract properties of the nodes, without altering the underlying tree structure inherited from the Isolation Forest. This design ensures that the additional overhead compared to the original IF algorithm remains minimal. Specifically, let L be a terminal node in a tree T within the IF ensemble. For each such node, the ALIF model maintains the following abstract properties:

- O_L : number of labelled anomalous points in the terminal region L ;
- I_L : number of labelled normal points in the terminal region L ;
- $\bar{h}(L)$: corrected depth of node L

To enable updates in the IF model, we redefine the anomaly score so that it incorporates the acquired labels. In ALIF, the anomaly score for a point x is computed in the same way as in the standard IF, but with the node depth function h replaced by the abstract property $\bar{h}$. The anomaly score is therefore given by:

$$a(x) = 2^{-\frac{E[\bar{h}(L_x)]}{c(\psi)}}. \quad (3.1)$$

The model begins with a fully unsupervised initialization, where both O_L and I_L are set to zero for every node, and $\bar{h}(L)$ is initialized to the observed depth $h(L)$. With this initialization, ALIF reduces to the standard IF whenever no labeled data are available. Importantly, the models predictions are no longer tied directly to the tree structure itself, but instead depend only on the abstract node properties.

3.3.2 Update Step

Once the initial unsupervised fit is completed, the iterative active learning process begins. At each iteration, a point $x \in X$ is selected, its true label is obtained, and the model is updated accordingly. Following the rationale of Isolation Forest, leaves containing anomalies should be located close to the root of the tree, while leaves dominated by normal points should appear deeper in the structure. To formalize this intuition, we introduce the color of a leaf L , defined as the relative proportion of anomalies and normal points it contains:

$$\bar{k}(L) = \frac{O_L - I_L}{O_L + I_L}, \quad (3.2)$$

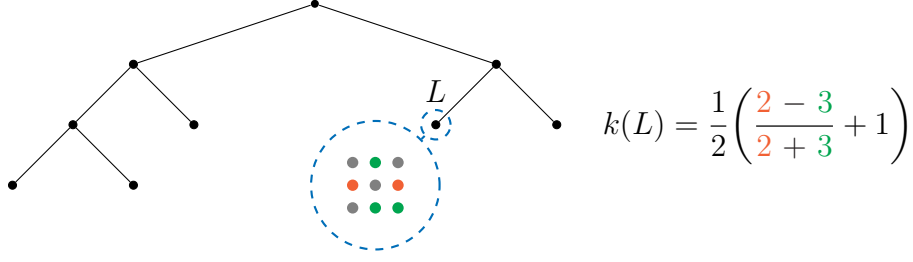


Figure 3.1: Whenever a new point is queried and its label is acquired, each iTree is examined and the leaves containing that point are identified. The color of a generic leaf L can then be visualized, where green dots denote labeled normal instances and red dots denote labeled anomalies. The corresponding value $k(L)$ is calculated according to Equation (3.3).

where I_L and O_L denote, respectively, the number of labeled normal instances and labeled anomalies in leaf L . The color thus characterizes the internal composition of a leaf, describing how labeled data are distributed within it. In particular, it provides a measure of how likely the leaf is to correspond to an anomalous region. Figure 3.1 illustrates the computation of leaf color. For convenience, we define a rescaled version of the color function that maps values into the closed interval $[0, 1]$:

$$k(L) = \frac{1}{2}(\bar{k}(L) + 1). \quad (3.3)$$

So that

- If L contains only normal points, then $k(L) \rightarrow 0$.
- If L contains only anomalies, then $k(L) \rightarrow 1$.

The supervised depth $h^s(L)$ is defined as a function of the leaf color. We consider two alternative formulations: the logarithmic supervised depth and the piecewise linear supervised depth. The logarithmic depth is given by:

$$h^s(L) = -c(\psi) \log_2(k(L)). \quad (3.4)$$

The logarithmic mapping in Equation (3.4) provides a natural way to transform the color $k(L) \in [0, 1]$ into positive real depths. By construction:

- as $k(L) \rightarrow 1$, then $h^s(L) \rightarrow 0$;

- when $k(L) = \frac{1}{2}$, then $h^s(L) = c(\psi)$;
- as $k(L) \rightarrow 0$, then $h^s(L) \rightarrow +\infty$.

However, this formulation suffers from instability due to the vertical asymptote at $k(L) = 0$, particularly when leaves are labeled exclusively with normal points. A simple workaround is to apply a threshold to the logarithm, saturating its growth and thereby mimicking the behaviour of a piecewise linear function.

To explicitly address this issue, we introduce the piecewise linear formulation defined as:

$$h^s(L) = \begin{cases} 2k(L)[c(\psi) - h_{max}] + h_{max}, & \text{if } 0 \leq k(L) < \frac{1}{2} \\ 2k(L)[h_{min} - c(\psi)] + 2c(\psi) - h_{min}, & \text{if } \frac{1}{2} \leq k(L) \leq 1 \end{cases} \quad (3.5)$$

where h_{max} and h_{min} denote, respectively, the maximum and minimum depths observed in the baseline IF during unsupervised training. in Equation (3.5). Here, the domain of $h^s(L)$ is restricted to the empirically observed range of tree depths. The mapping satisfies:

- as $k(L) \rightarrow 1$, then $h^s(L) \rightarrow h_{min}$;
- when $k(L) = \frac{1}{2}$, then $h^s(L) = c(\psi)$;
- as $k(L) \rightarrow 0$, then $h^s(L) \rightarrow h_{max}$.

Figure 3.2 illustrates the relationship between the leaf value $k(L)$ and the supervised depth $h^s(L)$ for both the piecewise linear and logarithmic formulations. It is straightforward to see that each mapping yields, in principle, a qualitatively different model. However model performance seems to be fairly insensible to the choice of mapping in practice.

3.3.3 Query Step

As is, ALIF can be seen as a weakly supervised algorithm for AD, where few labelled instances can be seamlessly incorporated into the baseline unsupervised model of IF. However, ALIF as the name suggests, is assumed to be incorporated into an active learning loop. Incorporating expert feedback into unsupervised anomaly detection aims to improve performance at a relatively low computational and labeling cost.

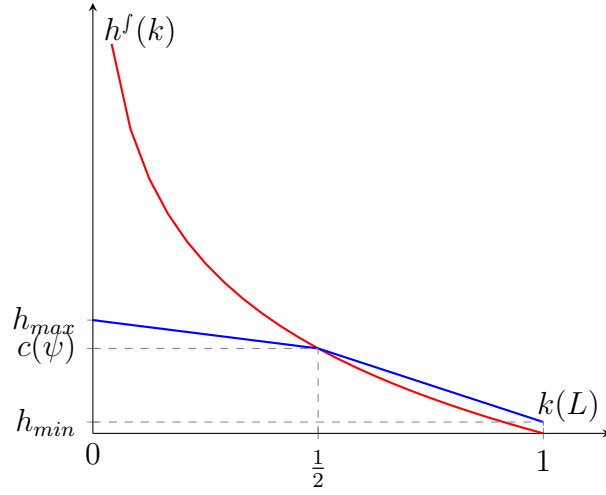


Figure 3.2: Visual comparison of the two possible definitions of supervised depth. The blue curve represents the piecewise linear formulation: when the leaf value $k(L)$ is close to 0 (i.e., the leaf is entirely populated by labeled normal points), the supervised depth of L approaches the maximum depth. Conversely, when $k(L)$ is close to 1 (i.e., the leaf is fully populated by anomalies), the depth approaches the minimum value. The red curve illustrates the logarithmic formulation: at the extremes $k(L) = 0$ and $k(L) = 1$, a threshold is applied, forcing the depth values to saturate at h_{max} and h_{min} , respectively.

A critical component of this process is the query strategy, i.e. the rule used to decide which points should be labeled. To do so, one must define a querying strategy, that is a process to decide what observations in the dataset are worth the extra labelling effort.

Identifying the optimal points to label, is no easy task in practice [KG20]. Beyond the mechanism that updates the Isolation Forest with new information, the success of the method depends heavily on how these points are chosen. Selecting the most informative queries can be decisive for the effectiveness of the approach. In the context of classical active learning, several query strategies have been studied [Set95].

Let X denote the pool of unlabeled candidate points, which may consist of the dataset originally used to train the IF or a new batch of data (e.g., collected during deployment). Using the supervised depth h_T^s , we define two query strategies in which the selection of points is based entirely on their supervised depth:

1. *Most anomalous*: at each iteration, the point selected to be labelled x^q is the point that has the highest anomaly score value in the current iteration, namely

$$x^q = \operatorname{argmax}_{x \in X} \left[\mathbb{E}_{T \in F} [h_T^s(x)] \right] \quad (3.6)$$

2. *Maximum uncertainty*: at each iteration the point selected to be labeled x^q is the one where the iTrees disagree the most, namely

$$x^q = \operatorname{argmax}_{x \in X} \left[\operatorname{std}_{T \in F} [h_T^s(x)] \right] \quad (3.7)$$

The first, following anomaly profiling, simply tries to maximize the probability of asking for labels of anomalous points, and achieves this by asking the label of the observation with the largest anomaly score. The rationale behind this strategy follows immediately from the previous considerations on class imbalance.

The latter strategy instead tries to provide information on observations where the model is uncertain. To achieve this, the authors ask for the label of the observation for which the observed height varies the most across individual trees in the ensemble. The rationale here is that, if there is variability in the ensemble, the model prediction is volatile, therefore the provided label can likely reduce this effect, biasing the model towards the correct prediction.

To ensure feasibility, each candidate point can be queried at most once. Once a label is acquired, it is assumed to be correct, and there is no need to request it again. From a business process perspective, querying the most anomalous point aligns with the role of a Decision Support System (DSS). Such a strategy highlights the instances considered most critical, prompting expert review. When the system flags a point as a strong anomaly, it triggers human validation, enabling the extraction of novel insights. This process combines domain expertise with computational analysis, leading to improved decision-making and efficient knowledge discovery, while reducing the effort required to sift through large volumes of data.

From a model-based perspective, a different strategy may be preferable: querying the point where the ensemble of iTrees is most uncertain. Here, expert input is directed toward the cases where the model struggles the most, ensuring that updates target the most critical ambiguities. However, given the typically unbalanced nature of anomaly detection datasets, this approach may result in labeling predominantly normal instances, limiting its practical usefulness despite its theoretical appeal.

3.4 Model complexity and overhead

We now turn to the computational complexity of predictions, updates, and the overhead introduced by ALIF relative to the underlying IF. A major advantage of the proposed update strategy is that it avoids retraining the entire forest. When new labels are acquired, ALIF updates only the color of the leaves containing the queried samples, leaving the rest of the structure unchanged. This can be achieved efficiently by storing I_L and O_L as leaf attributes. Consequently, the time complexity of each update is linear in the number of trees n_T , and, as in the original IF model, remains independent of the data dimensionality.

The model is explicitly designed for sequential updates. Incorporating a new labeled data point x requires modifying O_{L_x} and I_{L_x} for each tree in the ensemble. Since the maximum depth of any node is bounded by $\log_2(\psi)$, this operation takes $O(n_T \log_2(\psi))$ time. Notably, this is the same order of complexity as prediction in IF and remains constant with respect to the dataset size, with only small additional coefficients.

Similarly, to predict for a new point x , it suffices to retrieve the abstract property $\bar{h}$ from all terminal nodes containing x . This again requires $O(n_T \log_2(\psi))$ operations. Thus, compared to IF, the computational over-

head of ALIF is negligible, since in both cases the dominant cost arises from tree traversal. For the query strategy (Equations 3.7 and 3.6), the supervised depth must be evaluated across all candidate points in X . This procedure has the same complexity as performing inference on all of X , and, as with IF, is independent of the feature dimensionality.

Finally, in terms of memory requirements, ALIF introduces only a minor overhead. Each node must store two additional counters (I_L and O_L), but since these are scalar values, the increase is a small constant factor. Unless the baseline IF model is already heavily memory-optimized, this overhead remains practically negligible.

3.5 Experimental Results

In this section, we evaluate the performance of ALIF across all combinations of the proposed update strategies and query approaches. The goal is to provide a thorough assessment of the model, examining the efficiency of each configuration and offering practical guidance on their appropriate use.

We begin with synthetic datasets, including the particularly challenging shape illustrated in Figure 3.3, where normal data form a square toroid and anomalies are located inside it. On such datasets, Isolation Forest alone performs poorly, as it tends to isolate normal points more easily than anomalies. This limitation is evident in the initial iteration (corresponding to unsupervised training), where the model incorrectly assigns the highest anomaly score to a normal point in the bottom-left region of the toroid. By contrast, after 25 iterations, ALIF has successfully adapted to the task: it learns the correct decision function and achieves perfect classification of all points.

This improvement is also reflected in the second row of panels in Figure 3.3, which track the detectors performance across iterations using curves of different colors. As ALIF actively queries new points, both AUC and AP scores rise sharply, demonstrating how the integration of expert feedback enables rapid and consistent performance gains.

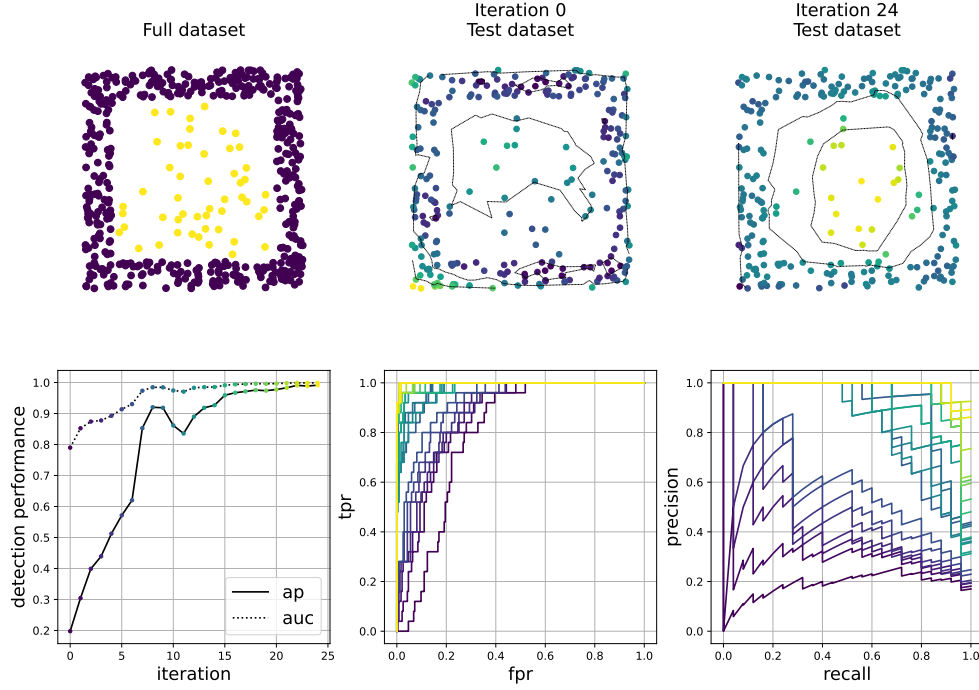


Figure 3.3: Evaluation of the algorithm on a square toroidal dataset. In this setting, anomalies are embedded inside a box surrounded by normal instances. Such a configuration is particularly challenging for Isolation Forest, since normal points (depicted in purple) are typically easier to isolate than anomalies (depicted in yellow). The second and third panels show the anomaly scores assigned to the test samples, with yellow indicating the highest scores and purple the lowest. The second row of panels reports the detectors performance across iterations: both the *area under the ROC curve* (AUC) and the *average precision* (AP), computed on the test set, increase rapidly, demonstrating the effectiveness of the proposed approach.

Dataset	Instances	Features	Anomalies	Contamination
AnnThyroid	7200	6	534	7.42%
Breastw	683	9	239	35 %
Cardio	1831	21	176	9.6%
Cover	286048	10	2747	0.9%
Ionosphere	351	33	126	36%
Letter	1600	32	100	6.25%
Mammography	11183	6	260	2.32%
Mnist	7603	100	700	9.2%
Optdigits	5216	64	150	3%
Pendigits	6870	16	156	2.27%
Pima	768	8	268	35%
Satellite	6435	36	2036	32%
Satimage-2	5803	36	71	1.2%
Thyroid	3772	6	93	2.5%
Vertebral	240	6	30	12.5%
Vowels	1456	12	50	3.4%
WBC	278	30	21	5.6%
Wine	129	13	10	7.7%

Table 3.1: Set of data used in the experimental phase. The first column gives the name of the dataset; the second column describes the number of instances contained in each set; the third column defines the total amount of features; the fourth column gives the number of outliers; the last column presents the contamination rates.

We further evaluated ALIF on a collection of 18 publicly available real-world datasets [She16a, DG17a], summarized in Table 3.1. These datasets span diverse domains such as medicine, industry, and the natural sciences, and vary considerably in size, dimensionality, and anomaly prevalence. In particular, the feature space ranges from 6 to 100 dimensions, while the contamination ratio, defined as the proportion of anomalies relative to the total number of samples, varies between 0.9% and 36%.

A key aspect of these datasets is how they were constructed. They are derived from multi-class classification tasks, reformulated for anomaly detection by designating one of the under-represented classes as anomalous. As a consequence, anomalies are not randomly scattered across the feature space,

as is often assumed in generic anomaly detection benchmarks. Instead, they occupy structured and domain-specific regions that can be difficult to identify without labels. This makes them particularly challenging for purely unsupervised methods, while offering an ideal setting to demonstrate the advantages of weakly supervised approaches.

In this context, ALIF leverages its hybrid nature: it begins with an unsupervised initialization and progressively improves as expert feedback is incorporated, effectively combining the flexibility of unsupervised detection with the precision of supervised learning.

To quantify performance, we adopted the average precision (AP) score. Each dataset was split equally into training and testing sets. For every experiment, the algorithm was allowed a maximum of 25 queries, and 50 independent runs were conducted to capture the distribution of performance. The query budget was deliberately limited to 25 for two reasons: (i) in realistic scenarios, labeling is costly and querying more points quickly becomes impractical; and (ii) beyond this threshold, one may enter a regime where fully supervised approaches become more natural. For transparency and reproducibility, an open-source implementation of ALIF is available at github.com/tombarba/ActiveLearningIsolationForest.

We first evaluated the four possible combinations of the two proposed update strategies with the two query strategies, in order to identify the configuration that performs best across the majority of datasets. The results are reported in Figure 3.4. The initial point of each curve corresponds to the performance of the fully unsupervised Isolation Forest. As shown, all four combinations consistently outperform the baseline, demonstrating the benefit of incorporating even limited feedback. In the figure, solid lines represent the most anomalous query policy, while dashed lines correspond to the maximum uncertainty policy.

Although no single strategy dominates across all datasets, the piecewise linear update consistently provides more stable improvements. This observation is aligned with the discussion in Section 3.3, where we noted the instability of the logarithmic update. With regard to the query policies, the most anomalous approach generally achieves faster performance gains. This result is particularly appealing in the context of Decision Support Systems (DSS), since points with high anomaly scores would naturally raise alarms and be examined by experts regardless. In fact, this strategy is the one employed in the Active Anomaly Discovery (AAD) paradigm. For these reasons, in the following experiments we adopt the combination of most anomalous querying

with the piecewise linear update strategy as the default ALIF configuration, and use it for comparison against baseline and state-of-the-art competitors.

Since ALIF is an active learning extension of the classical Isolation Forest, we compared it primarily with other tree-based models: a fully supervised approach, Random Forest (RF), and a weakly supervised one, Isolation Forest with Active Anomaly Detection (IF-AAD). To broaden the comparison, we also included a structurally different method, ν -SSVM [LBST21b], a semi-supervised approach based on support vector machines rather than trees.

Random Forest serves as a natural baseline, being a widely known yet simple classification algorithm. However, it requires labeled samples from both inliers and outliers, and must be retrained whenever new labels are added, making it computationally demanding. Moreover, RF lacks an intrinsic query mechanism; thus, in our experiments, it was supplied with the same labels selected by ALIF.

Likewise, ν -SSVM must be retrained entirely to incorporate additional labels, as it relies on solving a quadratic programming optimization problem that scales poorly with dataset size. This limitation makes the method unsuitable for very large datasets; for instance, we were unable to run it on the *cover* dataset. As with RF, ν -SSVM has no built-in querying strategy, and so it was trained using the labels provided by ALIF.

By contrast, IF-AAD is specifically designed to integrate user feedback in an active manner, without requiring labels from both classes. Its publicly available implementation allowed us to include it as a strong state-of-the-art benchmark.

Figure 3.5 summarize the benchmark results. ALIF consistently achieves superior performance with only a small number of labels, generally outperforming both IF-AAD and ν -SSVM. In contrast, RF struggles under the strong class imbalance, as it often receives labels from only one class even after many iterations. For example, in *breastw*, *satellite*, and *satimage-2*, the querying process failed to provide RF with both class labels within 25 iterations and across 50 independent runs, leading to persistently poor detection performance.

3.6 Conclusions

In this chapter, we introduced ALIF, a modification of the unsupervised Isolation Forest designed to improve its performance in anomaly detection

https://github.com/shubhomoydas/ad_examples

tasks. Since fully labeled datasets are rarely available, anomaly detection is typically performed with unsupervised models. However, because anomalies are highly domain-dependent, unsupervised models often struggle to detect them reliably, as their definition of anomaly may not align with that of the end user.

To address this issue, ALIF begins from an unsupervised baseline and iteratively adapts it toward the users definition of anomaly. This is achieved within an Active Learning framework, where the model selectively queries the most informative samples for labeling. By doing so, ALIF improves detection performance while minimizing the labeling effort, avoiding the need to fully annotate the training dataset. The model builds on two complementary steps: a query policy, which determines which points should be labeled, and an update policy, which integrates new information into the anomaly score computation. Our experiments on both synthetic and real datasets show that ALIF learns the target anomaly definition faster than competing methods, even under strict labeling constraints.

Beyond its baseline utility, ALIF offers promising opportunities for handling evolving data. In particular, it could help mitigate the effects of concept drift, where the very notion of what constitutes an anomaly changes over time. For example, conditions initially perceived as anomalous may later become acceptable and no longer worth monitoring. In such cases, ALIF can smoothly adapt the detector in response to updated expert feedback. On the other hand, the current approach is not directly equipped to handle data drift, i.e., shifts in the distribution of the input features. Addressing this limitation may require integrating ALIF with continuously learning tree-based structures such as Mondrian Forests [LRT14, MGS⁺20].

Finally, ALIF relies on two core assumptions: (i) users or human operators are available to provide labels, and (ii) these labels are accurate. While the first assumption is realistic in decision support systems, the second is less robust. Experts may make mistakes or lack full visibility of complex systems, resulting in noisy or partially incorrect labels. In future work, we plan to relax this assumption by extending ALIF into a broader framework capable of accommodating erroneous feedback to some extent, further increasing its practical applicability.

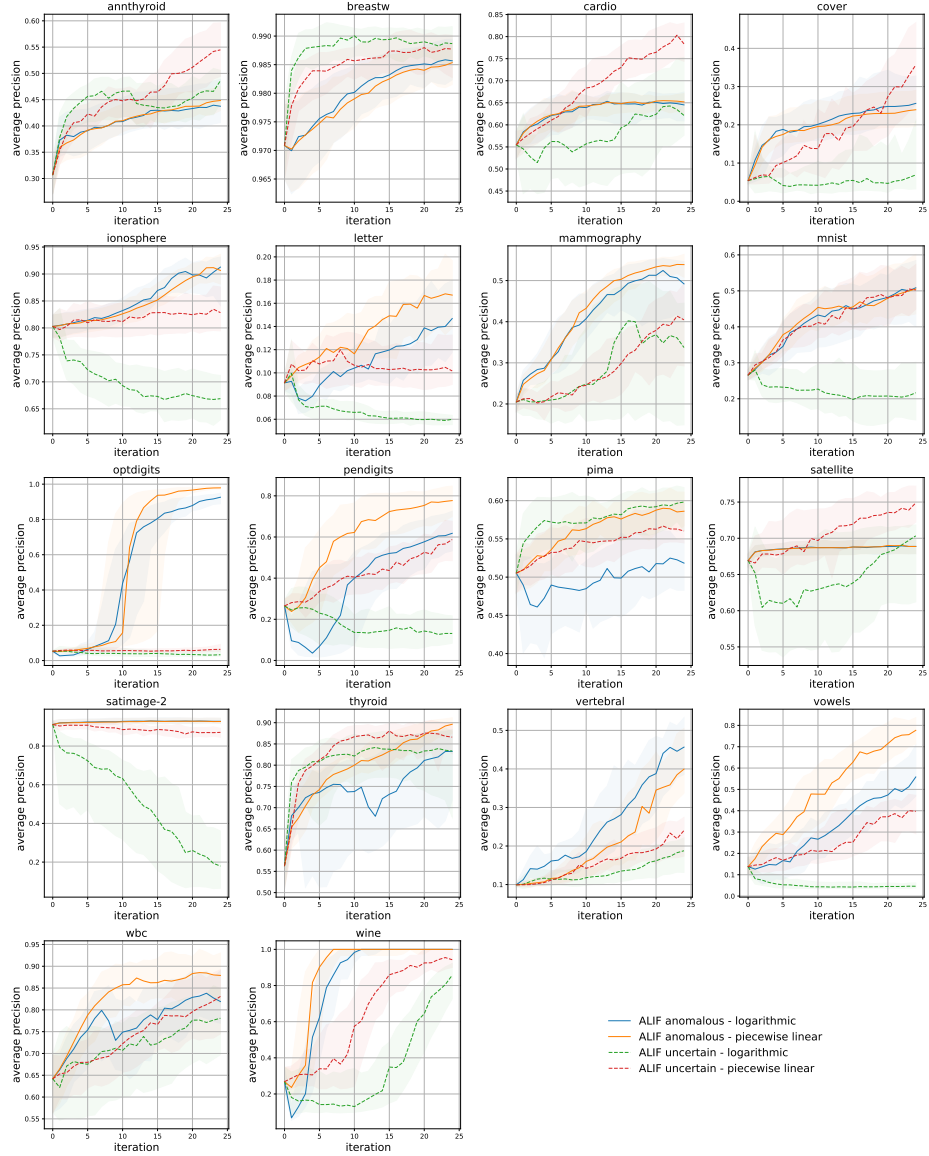


Figure 3.4: Performance of the four combinations of the proposed strategies across the datasets in Table 3.1. Overall, querying the most anomalous point emerges as the most effective choice, typically yielding faster performance improvements. Regarding the update strategies, both the logarithmic and the piecewise linear depths provide reasonable results, although the linear formulation shows moderately greater stability.

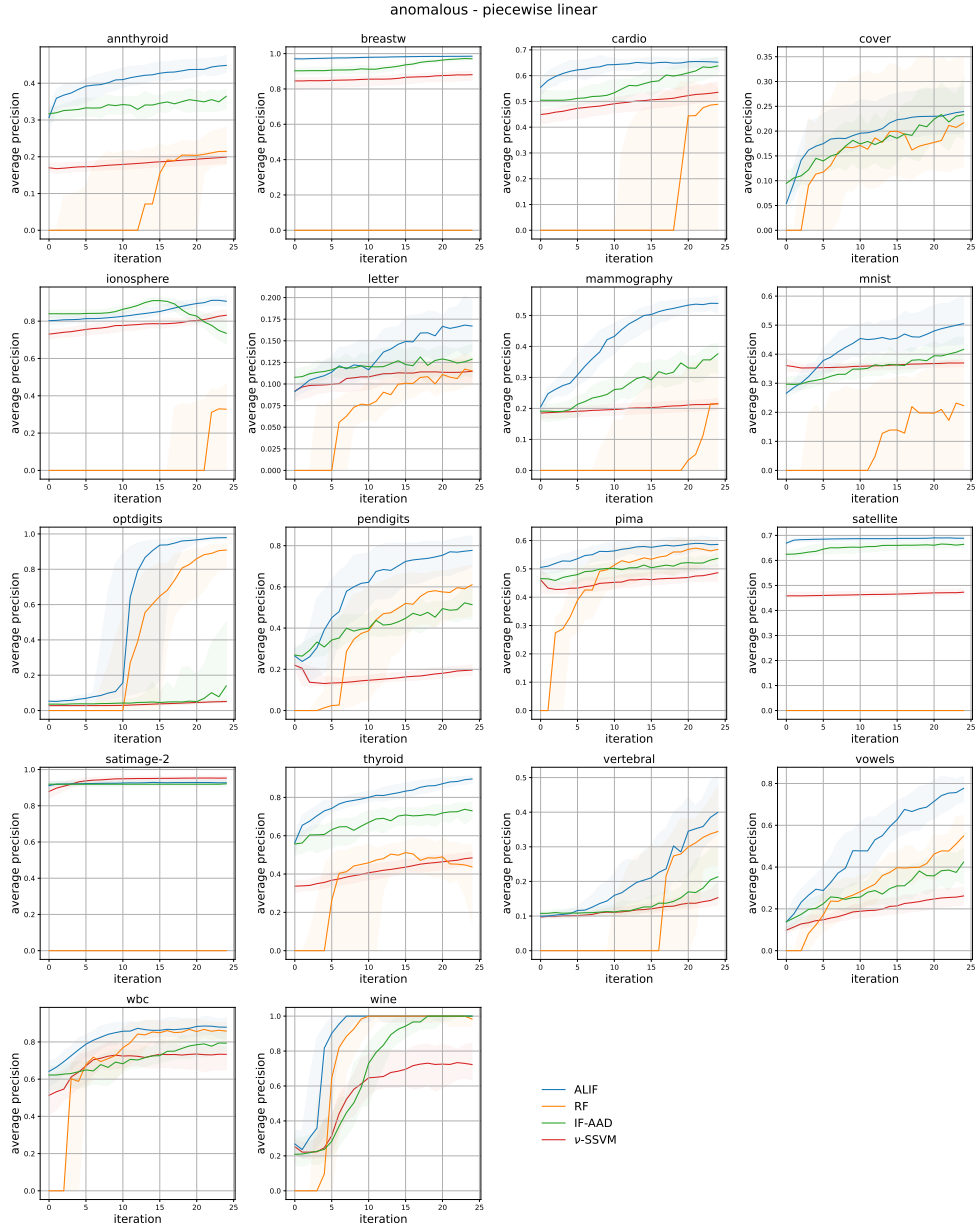


Figure 3.5: Comparison of most anomalous - piece-wise linear ALIF with IF-AAD, ν -SSVM and RF. It can be observed that, in general our method represents the best course of action, having the highest performance score usually with a very small amount of labeled data. Due to the computational complexity of ν -SSVM it was not feasible to test the *cover* dataset.

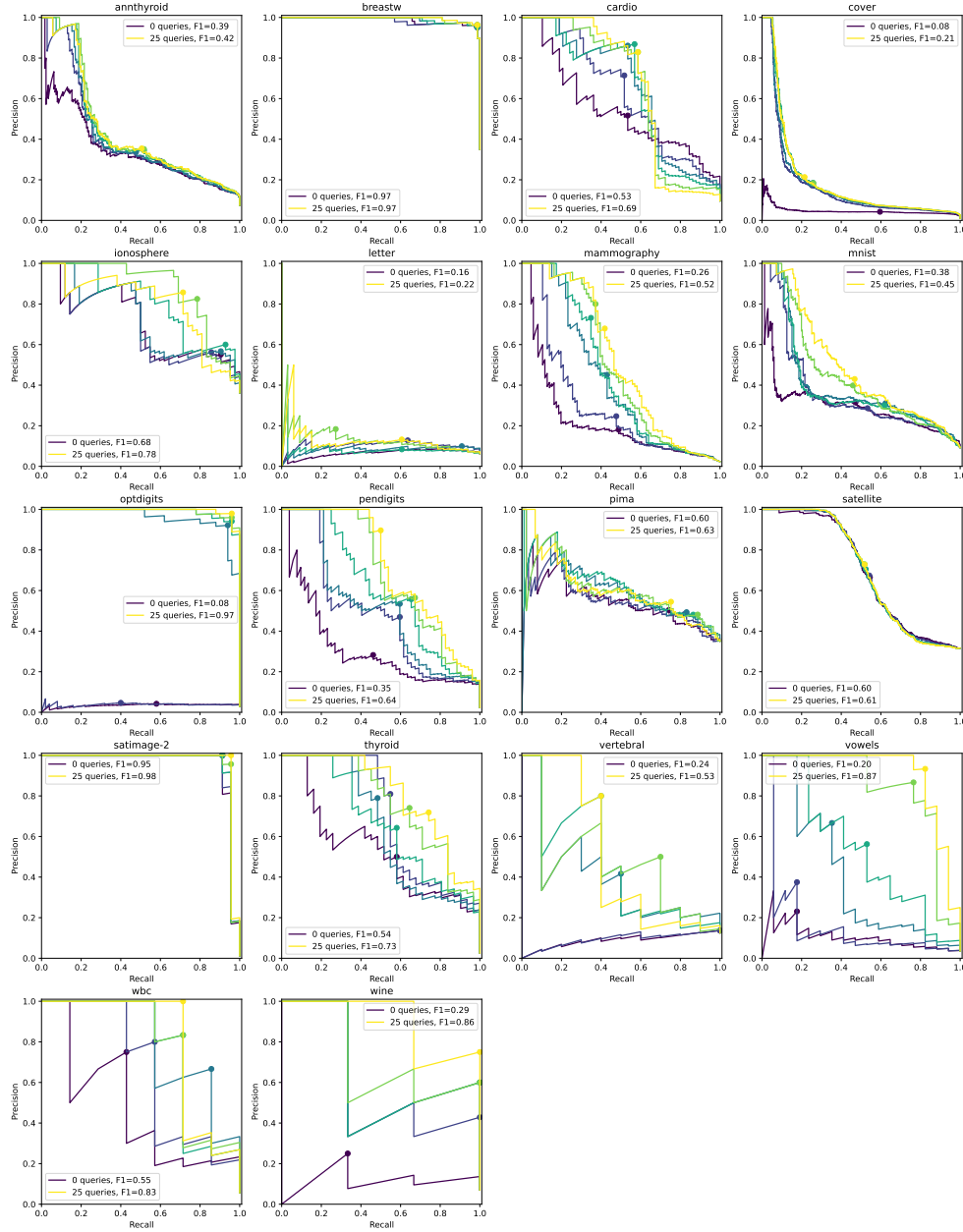


Figure 3.6: Detailed performance evolution of ALIF (one random run). The line plots show the precision recall curve color coded with the number of queries performed. Solid dots highlight the optimal threshold maximizing the F1 score.

4

Bayesian Active Learning Isolation Forest

4.1 Introduction

Due to the high cost of labelling large datasets, unsupervised Anomaly Detection models have been widely applied in many applications. In many scenarios, Anomaly Detection is typically embedded in Decision Support Systems, where users can monitor complex systems with increased efficiency. However, due to the subjective or domain-specific definition of anomalies, unsupervised models tend to perform poorly as they are based on general definitions of anomaly that might not coincide with the end-user expectations. In many cases human tagging of small amount of data can be feasible: For such scenarios, strategies to properly embed such information in the model and to guide the tagging may be extremely useful. Indeed to solve this problem, scientists are proposing models able to interactively query the end-user, training the detector towards the expected anomaly definition while reducing the human labelling effort; such scenario is especially feasible when Decision Support Systems are in place, since users have simple ways to interact with the Anomaly Detection modules.

In this context, a novel Active Learning algorithm is here proposed, called Bayesian Active Learning Isolation Forest (BALIF), that, starting from the solution provided by a popular detection model named Isolation Forest, it is able to query the user and update by means of a Bayesian strategy. BALIF employs the unsupervised solution provided by the Isolation Forest as its prior, and then improves its detection capabilities as the user interacts with it. Experimental results on real datasets show that, with a limited human effort, the proposed approach is able to learn faster than other state-of-art weakly supervised and supervised approaches.

The Active Learning Isolation Forest (ALIF)[MBS22] algorithm represents the starting point for this work. Chapter 3 is entirely dedicated to a discussion of the key details of this work. The ALIF model is specifically designed for Active Learning, being able to update efficiently by incorporating expert feedback. The core contributions of this work, being:

- an alternative theoretical formulation of the ALIF model, that leverages basic probabilistic programming notions to re-frame the existing algorithm from a Bayesian point of view.
- the possible extensions that come natural under the new interpretation, as well as the advantages brought forward by the richer description.

While ALIF presents one of the first approaches in the literature to deal with active learning with Isolation Forest, it has some major weaknesses that the approach presented in this work tries to address. Indeed even if ALIF works well in practice, it is completely based on heuristics. Moreover due to its simple updating procedure that does not take into account model uncertainty, it may have an unstable learning process and in presence of dense anomalous areas it suffers of slow learning rates that worsen the algorithm data efficiency. The proposed approach, named BALIF, tries to improve on ALIF by formalizing its heuristic rules using a Bayesian framework and therefore proposing a generalized approach that is more statistically grounded and reliable.

4.2 Bayesian Inference

In the field of Active Learning (AL), when the underlying models are probabilistic, the integration of newly acquired labels often requires performing an inference step. This is a broad and technically rich topic; here we restrict

ourselves to a brief introduction of the key ideas needed for later sections. For a more detailed treatment, the reader is referred to [Bar12].

Let θ denote the parameters of a model, with prior beliefs over θ encoded by a probability distribution $p(\theta)$. When new data d are observed, and assuming knowledge of the data-generating process through the likelihood $p(d \mid \theta)$, the natural question is how to update $p(\theta)$ to reflect this new evidence. Applying Bayes theorem, the posterior distribution is given by:

$$p(\theta \mid d) = \frac{p(d \mid \theta) p(\theta)}{\int p(d \mid \theta) p(\theta) d\theta} \propto p(d \mid \theta) p(\theta). \quad (4.1)$$

A major challenge in practical applications lies in computing the evidence term, $\int p(d \mid \theta) p(\theta) d\theta$, which ensures normalization. While this is merely a constant with respect to θ , evaluating the integral is often analytically intractable. Closed-form solutions exist only in special cases.

The most common tractable scenario arises when the prior and posterior belong to the same family of distributions, in which case the prior is called a conjugate prior for the likelihood. Choosing a conjugate prior simplifies Bayesian updating substantially. All members of the exponential family of distributions admit conjugate priors. For example, the Normal distribution is self-conjugate, while the Beta distribution is conjugate to both Bernoulli and Binomial likelihoods. This property will be central in what follows.

The Beta distribution is typically parameterized as:

$$p(\theta) \sim \text{Beta}(\alpha, \beta) \propto \theta^{\alpha-1} (1 - \theta)^{\beta-1}. \quad (4.2)$$

Alternatively, it can be expressed in terms of its mean $\mu = \frac{\alpha}{\alpha+\beta}$ and sample size $\nu = \alpha + \beta$:

$$p(\theta) \sim \text{Beta}(\mu, \nu) \propto \theta^{\mu\nu-1} (1 - \theta)^{(1-\mu)\nu-1}. \quad (4.3)$$

A key advantage of conjugacy is computational efficiency: Bayesian updating reduces to simple parameter updates, requiring only basic arithmetic rather than integration or approximate sampling.

In particular, consider a Beta prior $p(\theta) \sim \text{Beta}(\alpha, \beta)$ combined with a Binomial likelihood $p(k \mid \theta) \sim \text{B}(n, \theta)$. After straightforward algebra, the posterior becomes:

$$\begin{aligned} p(\theta \mid k) &\propto p(\theta) p(k \mid \theta) \\ &\propto \theta^{\alpha-1} (1 - \theta)^{\beta-1} \theta^k (1 - \theta)^{n-k} \\ &\sim \text{Beta}(\alpha + k, \beta + n - k). \end{aligned} \quad (4.4)$$

The same relation holds for a Bernoulli likelihood, which can be seen as a Binomial with $n = 1$. In this case, the Bayesian update reduces to simple additive adjustments to the prior parameters, directly determined by the observed realization k .

In cases where a closed-form solution for the posterior is not available, the distribution can be approximated computationally. One straightforward option is to evaluate the evidence integral numerically. While conceptually simple, this approach quickly becomes infeasible as the dimensionality of the parameter space increases, due to the exponential growth in computational cost.

An alternative, and more widely used, approach is to approximate the posterior by sampling. In this context, one of the most common techniques is Markov Chain Monte Carlo (MCMC). MCMC methods generate a sequence of samples by constructing a Markov chain, where each new sample depends only on the previous one. If the chain is designed appropriately, its limiting distribution converges to the target posterior. After a sufficient number of iterations, the chain reaches a stationary regime, and the collected samples can be treated as draws from the posterior distribution.

Although MCMC and related numerical approximation methods are powerful, a detailed discussion of these techniques lies outside the scope of this work. In the following, we focus primarily on cases where conjugate priors can be employed, allowing us to derive explicit update rules and avoid the computational overhead of sampling-based methods.

4.3 Bayesian Active Learning Isolation Forest

The central idea behind ALIF is both simple and effective: by attaching abstract information to the nodes, the efficiency of the Isolation Forest tree structure can be preserved while still allowing the model to adapt its predictions with minimal computational overhead. However, ALIF is primarily grounded in intuition and heuristic rules. BALIF seeks to overcome this limitation by offering a principled theoretical foundation. Through a Bayesian interpretation, BALIF not only provides a clearer rationale for the mechanics of ALIF but also naturally suggests extensions and enhancements that arise from this perspective. In this sense, BALIF should be regarded as a general framework, with ALIF representing a particular instance within it.

4.3.1 Generalized abstract properties

In ALIF, the iterative updating of node parameters based on newly acquired labels is, in spirit, reminiscent of Bayesian updating with conjugate priors, as introduced in Section 4.2. This observation motivates a natural generalization: rather than storing deterministic, pointwise estimates of node information, we allow for uncertainty in these quantities by maintaining probability distributions over them. We refer to this extended framework as Bayesian-ALIF (BALIF).

To ground this idea, let us reconsider the task of anomaly detection and the role of anomaly scores. Given an observation x falling in a region L , the goal is to assess the probability that x is anomalous. In practice, models typically approximate this probability with an anomaly score $a(x)$, where higher scores are intended to reflect higher likelihoods of anomaly. However, these scores are not probabilities in a strict sense, and no effort is usually made to enforce such an interpretation.

From a Bayesian perspective, assuming anomaly scores correspond directly to probabilities is a simplifying but often useful assumption that enables principled inference. BALIF builds on this view by explicitly modeling the anomaly rate of a region L , i.e., the probability that a randomly drawn point in L is anomalous:

$$\text{Ar}(L) = \text{P}[x \text{ is anomaly} \mid x \in L] . \quad (4.5)$$

To represent this uncertainty, BALIF associates with each node an abstract property in the form of a probability distribution over $[0, 1]$, describing beliefs about $\text{Ar}(L)$. Formally, this distribution is denoted as:

$$g_L(z) = \frac{d}{dz} \text{P}[\text{Ar}(L) \leq z], \quad (4.6)$$

which corresponds to the probability density function of $\text{Ar}(L)$.

4.3.2 Node property updates

Since the abstract properties stored in each node are now probability distributions, updating the model dynamically as new labels are obtained becomes more challenging. To keep the problem tractable, we introduce the following simplifying assumptions:

- anomaly rates of terminal, non-overlapping regions are independent of each other, unless otherwise specified;

- anomaly rates of partially overlapping terminal regions across different trees are independent;
- labeled points are sampled independently of the anomaly rates of the regions they fall into.

These assumptions do not strictly hold in every scenario, but they allow for efficient computation. Their validity and possible relaxations will be discussed in the conclusion section. Under this formulation, each terminal region L maintains a distribution $g_L(z)$ over its anomaly rate. Incorporating a new data point x with label y consists of updating the belief about $\text{Ar}(L)$ for every region containing x . Since regions are assumed independent, this update can be carried out separately for each tree in the ensemble.

Formally, for any region L containing x , the posterior distribution is:

$$g_L(z \mid x \in L, y) = \frac{g_L(z), p(x \in L, y \mid z)}{p(x \in L, y)}. \quad (4.7)$$

Expanding the likelihood term, we obtain:

$$p(x \in L, y \mid z) = p(x \in L \mid z) p(y \mid x \in L, z) \quad (4.8)$$

$$= p(x \in L \mid z) z^y (1 - z)^{1-y}, \quad (4.9)$$

where the conditional distribution of y is Bernoulli with parameter z , the anomaly rate of region L . Under the assumption that point locations are independent of anomaly rates, the term $p(x \in L \mid z)$ cancels out, and the likelihood reduces to a Bernoulli distribution.

From Section 4.2, we know that the Bernoulli distribution admits a conjugate prior in the Beta family. Therefore, if $g_L(z)$ is modeled as a Beta distribution, the update step can be performed analytically in closed form. Specifically, let

$$g_L(z) \sim \text{Beta}(\alpha, \beta) \propto z^{\alpha-1} (1 - z)^{\beta-1}. \quad (4.10)$$

After observing a labeled point (x, y) , the posterior is:

$$g_L(z \mid x \in L, y) \sim \text{Beta}(\alpha + y, \beta + 1 - y). \quad (4.11)$$

In other words, the update rule simply increments α when the new point is labeled as an anomaly ($y = 1$), or β when it is labeled as normal ($y = 0$). Thus, for every leaf L , the abstract property g_L can be fully characterized by just two parameters, α_L and β_L . If the initial values are α_L^0 and β_L^0 , and O_L and I_L denote the number of observed anomalies and normal points respectively, then: $\alpha_L = \alpha_L^0 + O_L$ and $\beta_L = \beta_L^0 + I_L$.

4.3.3 Initialization and IF prior

Since in BALIF the abstract properties of nodes parameterize probability distributions, the initialization of α_L^0 and β_L^0 can be interpreted as the choice of a prior distribution for $g_L(z)$.

A natural starting point is to set an uninformative prior by choosing $\alpha_L^0 = \beta_L^0 = w_0$. For $w_0 = 1$, this yields a uniform distribution over $[0, 1]$. If, instead, we wish to bias the prior toward extreme values reflecting the belief that anomaly rates tend to be polarized, we can set $w_0 < 1$, with smaller values enforcing stronger polarization.

However, enforcing $\alpha_L^0 = \beta_L^0$ is not optimal. In fact, we are not starting from a completely uninformed perspective, since we can leverage the predictions of the underlying unsupervised Isolation Forest. This motivates the use of the IF output to set the prior mean.

To fully characterize the Beta prior, two degrees of freedom must be fixed: the mean (which can be constrained to match the IF prediction when no labeled data are available) and the sample size $\nu_L^0 = \alpha_L^0 + \beta_L^0$, which controls the priors strength relative to new observations.

In practice, ALIF disregards IF predictions as soon as even a single relevant label is obtained, and performs well empirically. This suggests that a weak prior is preferable, allowing the model to adapt rapidly as new labels are incorporated. Accordingly, we set ν_L^0 to its minimum possible value, subject to $\alpha_L^0 \geq w_0$ and $\beta_L^0 \geq w_0$. The idea is to add only the minimal number of fictitious observations ($\Delta\alpha = \alpha_L^0 - w_0$, $\Delta\beta = \beta_L^0 - w_0$) needed to shift the uninformative prior to one consistent with the IF prediction.

For example, if $w_0 = 1$ and we match the IF anomaly score $2^{-\frac{h(L)}{c(\psi)}}$ to the mean of g_L , we obtain:

$$\alpha_L^0 = \frac{2^{-\frac{h(L)}{c(\psi)}}}{\frac{1}{2} - \left| 2^{-\frac{h(L)}{c(\psi)}} - \frac{1}{2} \right|}, \quad \beta_L^0 = \frac{1 - 2^{-\frac{h(L)}{c(\psi)}}}{\frac{1}{2} - \left| 2^{-\frac{h(L)}{c(\psi)}} - \frac{1}{2} \right|}. \quad (4.12)$$

This strategy provides a direct extension of IF by reinterpreting its anomaly scores as prior means. However, an important limitation must be acknowledged: the anomaly scores of IF are not true probabilities and, in practice, do not span the full interval $(0, 1)$. Because of the truncation applied during tree construction, very low scores are rarely observed, even for points that are highly unlikely to be anomalous. As a result, priors constructed in this way are biased toward higher anomaly rates.

In this work, we mitigate this effect by linearly rescaling the IF score range to cover $(0, 1)$. Consequently, BALIF does not exactly reproduce the raw IF predictions, but rather a rescaled version. While this is not an entirely satisfactory solution, it serves as a pragmatic adjustment. A more principled treatment of this issue is deferred to the conclusions section.

4.3.4 Anomaly score

Similarly to ALIF, for a given point $x \in \hat{L}_x = \cap_T L_{x,T}$, BALIF defines its prediction as a function of the abstract properties stored in the leaves that contain x :

$$a(x) = f\left(g_{L_{x,T_1}}, \dots, g_{L_{x,T_i}}, \dots, g_{L_{x,T_{n_T}}}\right). \quad (4.13)$$

Thus, to produce a prediction for x , we must aggregate the distributions $g_{L_{x,T}}$ describing the models belief about the anomaly rate of each leaf. Importantly, if two points x_1 and x_2 belong to the same region across all trees, i.e., $\hat{L}_{x_1} = \cap_T L_{x_1,T} = \cap_T L_{x_2,T} = \hat{L}_{x_2}$, then BALIF will necessarily produce identical predictions for them. In other words, the most precise prediction achievable for any x depends on the anomaly rate of the joint region $\hat{L}_x$.

Consequently, the anomaly score for x can naturally be represented as a likelihood distribution $G_x(z)$ over the anomaly rate of $\hat{L}_x$, which may then be summarized by a single scalar value $a(x)$ if required. Different ways of combining the leaf-level distributions lead to different formulations of BALIF predictions.

In what follows, we describe two possible strategies: the *Likelihood* formulation and *Naive* formulation. The Likelihood assumes the strict condition $\text{Ar}(\hat{L}_x) = \text{Ar}(L_{x,T})$ for all T , i.e., that all trees provide consistent estimates of the anomaly rate. The Naive assumes $\text{Ar}(L_{x,T}) \approx \text{Ar}(L_{x,T} \setminus \hat{L}_x)$, effectively treating the anomaly rates of partially overlapping leaves as similar, and combining them more loosely.

Likelihood ensemble prediction This ensemble prediction method relies on the assumption that the true anomaly rate does not vary significantly in the neighborhood of a point x . Formally, this translates into imposing equality of anomaly rates across all terminal regions containing x and their intersection:

$$\text{Ar}(\hat{L}_x) = \text{Ar}(L_{x,T_1}) = \dots = \text{Ar}(L_{x,T_i}) = \dots = \text{Ar}(L_{x,T_{n_T}}). \quad (4.14)$$

Under this assumption, the likelihood of $\text{Ar}(\hat{L}_x)$ can be expressed as the

product of the distributions associated with the leaves containing x :

$$G_x(z) = p\left(\text{Ar}(\hat{L}_x)\right) \propto \prod_T p(\text{Ar}(L_{x,T})). \quad (4.15)$$

Since each $p(\text{Ar}(L_{x,T}))$ is modeled as a Beta distribution, $G_x(z)$ is itself proportional to a Beta distribution with parameters given by the sum of the leaf-specific parameters:

$$\begin{aligned} G_x(z) &\propto \prod_T z^{(\alpha_{L_{x,T}} - 1)} (1 - z)^{(\beta_{L_{x,T}} - 1)} \\ &\sim \text{Beta}\left(1 + \sum_T (\alpha_{L_{x,T}} - 1), 1 + \sum_T (\beta_{L_{x,T}} - 1)\right). \end{aligned} \quad (4.16)$$

This closed-form representation provides not only a compact characterization of $G_x(z)$ but also a simple way to compute the anomaly score $a(x)$. Specifically, $a(x)$ can be obtained as the mode of the Beta distribution:

$$a(x) = \underset{z}{\text{argmax}}; G_x(z) = \frac{\sum_T (\alpha_{L_{x,T}} - 1)}{\sum_T (\alpha_{L_{x,T}} + \beta_{L_{x,T}} - 2)}. \quad (4.17)$$

This result is particularly elegant: it reduces the prediction for x to a simple ratio of aggregated anomaly counts across all terminal regions containing x . Importantly, this formulation requires no iterative optimization as predictions are obtained directly in closed form.

Naive ensemble prediction The second ensemble prediction strategy relies on a different assumption: for a given point x , the intersection of its terminal regions, $\hat{L}x = \cap_T Lx, T$, is much smaller than each individual $L_{x,T}$. Consequently, the anomaly rate of the residual region $L_{x,T} \setminus \hat{L}x$ is approximately equal to that of Lx, T , i.e.,

$$\text{Ar}(L_{x,T} \setminus \hat{L}x) \approx \text{Ar}(Lx, T). \quad (4.18)$$

Under this assumption, a straightforward algebraic argument leads to the following relation:

$$\text{Ar}(\hat{L}x) \approx \frac{1}{n_T} \sum_T \text{Ar}(Lx, T). \quad (4.19)$$

While the exact distribution of the average of n_T Beta-distributed random variables does not admit a closed form, for sufficiently large n_T it can be well

approximated by another Beta distribution. Matching the first two moments of the distribution yields:

$$G_x(z) \sim \text{Beta}(\mu, \nu), \quad (4.20)$$

with parameters:

$$\mu = \frac{1}{n_T} \sum_T \frac{\alpha_{L_x,T}}{\alpha_{L_x,T} + \beta_{L_x,T}}, \quad (4.21)$$

$$\nu = \sum_T (\alpha_{L_x,T} + \beta_{L_x,T}). \quad (4.22)$$

Using this approximation for $G_x(z)$, the anomaly score of x can be naturally defined as the expected probability of x being anomalous under the distribution of $\text{Ar}(\hat{L}_x)$:

$$\begin{aligned} a(x) &= \mathbb{E} [\text{Ar}(\hat{L}_x)] \\ &\sim G_x [\text{P}[; x \text{ is anomaly} \mid \text{Ar}(\hat{L}_x)]] \\ &= \frac{1}{n_T} \sum_T \frac{\alpha_{L_x,T}}{\alpha_{L_x,T} + \beta_{L_x,T}}, \end{aligned} \quad (4.23)$$

which coincides with the mean of G_x .

4.3.5 Connections with ALIF

In this section, we highlight the connections between the ALIF algorithm and its Bayesian extension, BALIF. In fact, the parameters used in BALIF are closely related to the abstract properties defined in ALIF, and the update rules of the two algorithms are effectively identical.

Specifically, since $\alpha_L = \alpha_L^0 + O_L$ and $\beta_L = \beta_L^0 + I_L$, the two formulations coincide in the limit where the prior weight $w_0 \rightarrow 0$. In this setting, the leaf color defined in ALIF corresponds exactly to the mean of the belief distribution in BALIF:

$$k_L = \frac{O_L}{I_L + O_L} = \text{mean}(\text{Beta}(\alpha_L, \beta_L)) = \frac{\alpha_L}{\alpha_L + \beta_L}. \quad (4.24)$$

Thus, ALIF can be interpreted as a special case of BALIF, where (i) labeled data are assumed to be sampled independently of regional anomaly rates, (ii) the prior weight is set to zero, and (iii) ensemble predictions reduce to the geometric mean of log-scores across trees.

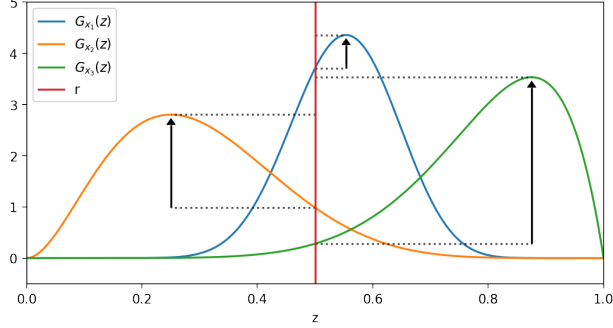


Figure 4.1: Example visualizations of likelihood margins for different prediction functions. The plots illustrate how the proposed margin metric captures not only the location of the likelihood peak but also the overall dispersion of the distribution. A sharply concentrated likelihood around the decision boundary produces a small margin, reflecting high model uncertainty, whereas a well-localized distribution near the extremes yields a large margin, indicating confident predictions.

For instance, consider the logarithmic variant of ALIF. Its prediction function for a point $x \in \hat{L}_x$ can be rewritten in terms of BALIFs belief distributions as:

$$a(x) = 2^{-\frac{\mathbb{E}[-c(\psi) \log_2(k_L)]}{c(\psi)}} = \left(\prod_{i=1}^{n_T} k_{L_x, T_i} \right)^{1/n_T} = \left(\prod_{i=1}^{n_T} \frac{\alpha_{L_x, T_i}}{\alpha_{L_x, T_i} + \beta_{L_x, T_i}} \right)^{1/n_T}. \quad (4.25)$$

This formulation makes explicit how ALIFs heuristic update and scoring rules naturally emerge as a degenerate case of BALIF under Bayesian reasoning.

4.3.6 Query strategy

When it comes to query strategies, BALIF allows us to build directly on the considerations made for ALIF. In particular, actively querying the points with the highest anomaly scores remains a natural and effective approach to improve detection performance. However, the Bayesian reinterpretation also enables us to propose an alternative to the classical uncertainty-based strategy.

In ALIF, model uncertainty was estimated by measuring disagreement across the ensemble of trees. While this idea can be carried over unchanged to BALIF, the Bayesian framework naturally provides a richer and more prin-

ciplined notion of uncertainty, namely, the likelihood distribution $G_x(z)$ associated with each point. Since model predictions are inherently represented as full likelihood functions (and only later condensed into point estimates when an anomaly score is required), we can directly assess how concentrated or diffuse the models beliefs are around a given prediction.

Intuitively, if the likelihood distribution is flat, the model is uncertain about the true anomaly rate. Furthermore, the location of this uncertainty matters: if the likelihood mass is near the decision boundary, even small fluctuations in the mode can lead to qualitatively different predictions. By contrast, if the likelihood is sharply concentrated near the extremes (close to 0 or 1), additional information is unlikely to alter the outcome significantly.

To exploit this insight, we introduce a new query strategy based on the likelihood ratio between the most likely value and a fixed reference point r :

$$\text{margin}_r(x) = \frac{\max_z G_x(z)}{G_x(r)} . \quad (4.26)$$

This ratio can be interpreted as comparing the likelihood of the models most confident prediction to that of the decision boundary defined at r . A natural uninformative choice is $r = 0.5$, which corresponds to the standard mid-point threshold between classes. Notably, when $r = 1$, selecting points with the largest margin reduces exactly to the most anomalous strategy.

Intermediate values of r interpolate between these two extremes: the margin-on- r strategy explicitly targets points whose predictions remain ambiguous in the vicinity of r , thereby prioritizing labels that help the model sharpen its decision boundary.

4.3.7 Model complexity and overhead

When analyzing the computational complexity of BALIF and the potential overhead it introduces, one might initially expect a significant increase relative to ALIF. However, in practice, all of the considerations discussed in Section 3.4 remain valid. Indeed, since ALIF can be interpreted as a special case of BALIF, the generalized formulation inherits essentially the same parameterization and update rules. The only substantive difference lies in the interpretation of these parameters: in BALIF they correspond to probability distributions rather than deterministic quantities.

The practical deviations between ALIF and BALIF are therefore limited to two aspects: (i) parameter initialization and (ii) ensemble prediction func-

tions. Importantly, neither of these introduces meaningful additional overhead. Regardless of the chosen strategy, each parameter can be initialized in constant time, requiring only a straightforward rescaling of the unsupervised IF predictions. The cost of this step is negligible when compared to the computational effort of training the underlying Isolation Forest.

While it is true that more elaborate formulations could, in principle, increase computational demands, the definitions considered in this work retain the same order of complexity as ALIF and the original IF. Specifically, the cost of inference remains $O(n_T \log_2(\psi))$, reflecting the traversal of the n_T trees in the ensemble.

Thus, BALIF preserves the key efficiency properties of ALIF, ensuring that the probabilistic generalization does not compromise the lightweight and scalable nature of the original framework.

4.4 Results

In this Section we present some experimental results both on synthetic data sets and on real world data. We follow a similar testing procedure to the one employed in the original ALIF paper, giving in depth considerations on synthetic tests, and introducing an optimization for the anomalous querying strategy. We then show real world performance of the AL loop comparing across BALIF variants and to ALIF.

4.4.1 Experiments on synthetic dataset

Following the experimental setup of the original ALIF paper, we evaluate the proposed algorithm on a synthetic dataset consisting of a two-dimensional square toroidal ring of densely packed normal points surrounding a sparse cloud of anomalies at the center (Figure 4.4.1). This configuration is particularly challenging for the IF algorithm: anomalous central points are harder to isolate, whereas normal points near the boundary of the domain are often separated more quickly.

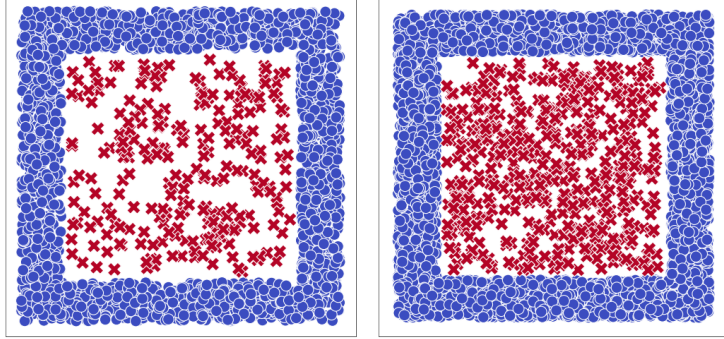


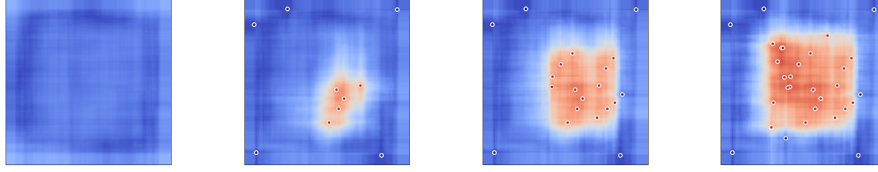
Figure 4.2: Visualization of Square toroidal datasets: Sparse version (500 datapoints) on the left, Dense version (5000 datapoints) on the right.

In prior work, ALIF demonstrated its ability to rapidly correct the poor initial predictions of the underlying IF, reaching high performance after only a few iterations. In our experiments, however, we observe that both ALIF and BALIF are sensitive to changes in the hyperparameters of the data generation process. These effects depend strongly on the chosen querying strategy. To illustrate this, we present a set of representative scenarios and provide intuition for the observed behaviors.

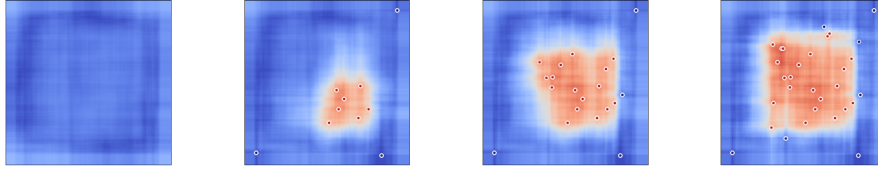
As a baseline, we consider the same setup as in the original paper: approximately 500 data points with a 10% anomaly rate, split evenly into training and test sets via stratified random sampling. Running simulations in this setting allows us to qualitatively describe how the algorithm behaves.

When querying the most anomalous points (Figure 4.3a), the model initially selects samples near the domain edges, reflecting the inaccurate predictions of the base IF. Through these queries, the algorithm quickly learns that the edges are unlikely to be anomalous. Shortly thereafter, it begins to query points in the central region, eventually discovering its first true anomaly. Once this occurs, the model infers that nearby points are also likely anomalous, initiating a propagation effect: the region of anomalous predictions expands with each new anomaly confirmed, until the anomalous cluster is fully delineated.

For this choice of parameters, the process typically requires on the order of 30-40 queries to complete, depending on how quickly the first anomaly is discovered and how efficiently subsequent queries refine the region boundary. Importantly, this behavior highlights the dependence of performance on the density of the anomalous region.



(a) *Anomalous* query strategy

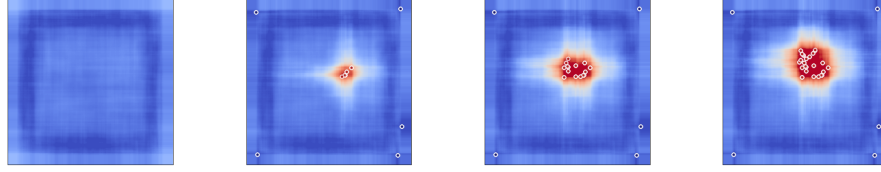


(b) *Margin* query strategy

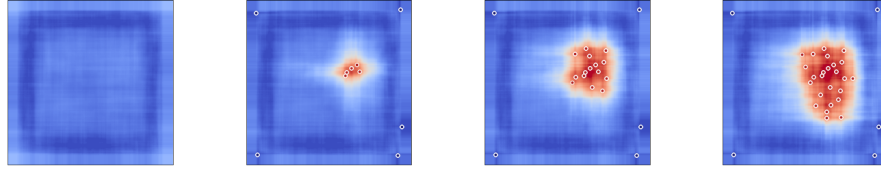
Figure 4.3: Square toroidal datasets and qualitative evolution of BALIF predictions on the synthetic "square toroid" dataset after 0, 10, 20 and 30 queries. Low density experiment.

If we increase the dataset size by an order of magnitude, making it ten times denser while keeping the same structure, we observe that the algorithm requires substantially more iterations to fully delineate the anomalous region (Figure 4.4.1). This behavior can be intuitively explained: the speed at which the model profiles an anomalous cluster is inversely proportional to the density of points in that region. Consequently, in denser settings, the discovery process is significantly slowed down.

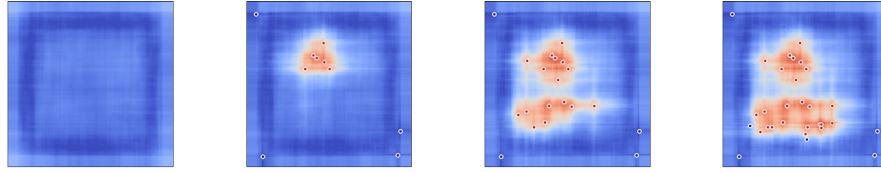
A straightforward way to mitigate this effect is to introduce a subsampling step. Instead of allowing the model to query labels from the entire dataset, we restrict its choices to a random subset of candidate points. This modification breaks the dependency on density: the model can no longer progress in small incremental steps, but is instead encouraged to make larger jumps across the feature space, which accelerates the identification of anomalous regions.



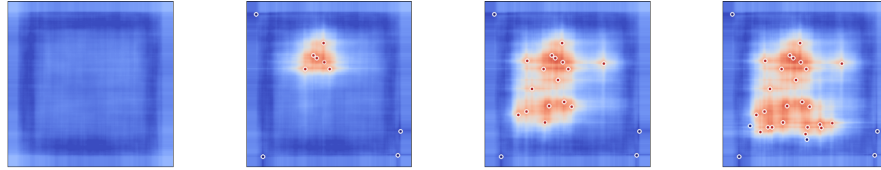
(a) *Anomalous* query strategy



(b) *Margin* query strategy



(c) *Anomalous* query strategy + subsample



(d) *Margin* query strategy + subsample

Figure 4.4: Qualitative evolution of BALIF predictions on a *dense* version of the synthetic "square toroid" dataset after 0, 10, 20 and 30 queries.

In addition, subsampling reduces the computational burden of the update step, since the model no longer needs to be evaluated on every data point. This property is particularly attractive in online or streaming scenarios, where efficiency and responsiveness are critical.

When switching to the margin query strategy, the dependence on data density is largely mitigated (Figure 4.4b). Under this approach, the model

ignores labels from points close to already queried ones, unless they lie near a boundary between anomalous and normal regions.

Although generally more robust, the margin strategy is highly sensitive to the choice of threshold parameter r . If r does not properly capture the region of uncertainty, the algorithm may spend a considerable portion of its labeling budget refining estimates of anomaly rates within regions, without substantially improving the classification task. This sensitivity is amplified by any bias introduced through the priors inherited from the underlying IF, as such bias effectively shifts the uncertainty region and can reduce the effectiveness of the strategy.

In summary, both query strategies present advantages: the most anomalous strategy is simple, efficient, and well aligned with decision support scenarios, while the margin strategy offers greater robustness in dense datasets but requires careful tuning. Their relative effectiveness ultimately depends on the structure and characteristics of the data.

4.4.2 Experiments on real world data sets

At its core, most tasks in anomaly detection can be interpreted as a binary classification problem, even if highly unbalanced, where anomalies need to be separated from normal data instances. Even when the model does not directly classify points into anomaly or inlier, but yields for instance an anomaly score, it generally needs to be converted into a classifier by finding an appropriate threshold. Unfortunately, as discussed in [SR15], when the dataset is highly unbalanced like in anomaly detection settings, true positive rate and false positive rate are not reliable metrics, so in this work we will use precision and recall to measure the detection performances. In particular, to obtain an unique score that assess the detection performances independently on the threshold we will use the average precision score.

The datasets used in our experiments are taken from the ODDS repository [She16b]. They originate from multi-class classification problems, where one underrepresented class is treated as anomalous and the remaining classes represent normal data. Consequently, anomalies in these datasets are not arbitrary out-of-distribution points, but instead occupy specific regions of the domain and exhibit structured patterns. This makes them particularly challenging to address in a fully unsupervised setting.

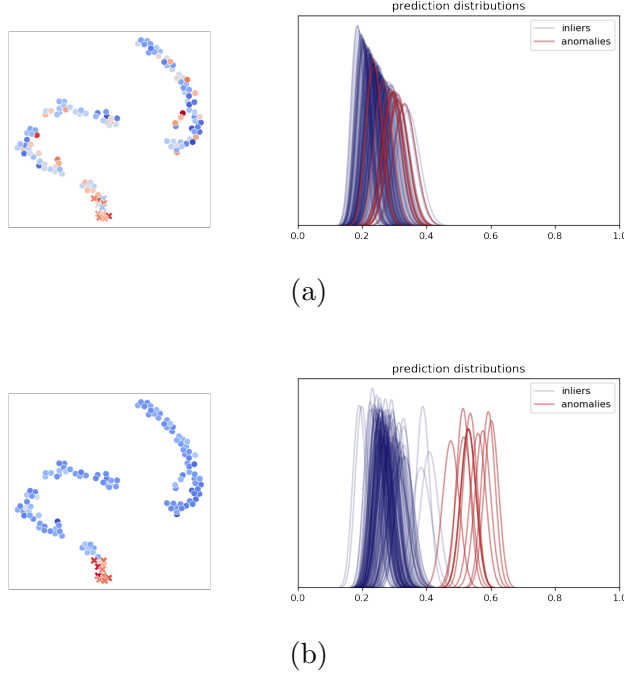


Figure 4.5: Comparison of BALIF predictions (left) and belief distributions (right) on "wine" dataset without labels (top), and after 10 queries (bottom). Results are qualitatively and quantitatively similar amongst different model configurations (see Figure 4.7).

To minimize potential bias from train-test splits (as discussed in the previous section), we adopt an 80-20 split for all datasets. Moreover, at each iteration the model can only query points from a random subset of the original training set, with a maximum size of 512. This subset is resampled at every iteration.

To provide a qualitative illustration of the BALIF algorithms performance, Figure 4.5 shows results on the wine dataset before and after 10 iterations using the anomalous querying strategy. The left plots display a two-dimensional t-SNE projection of the dataset, where points are colored by their anomaly scores and true anomalies are marked with crosses. The right plots show the ensemble prediction distributions for each point. Without supervision, predictions are poor; however, after 10 iterations, the true anomalous points become much more clearly separated.

In our experiments we were guided by three main research questions:

1. In a weakly supervised scenario where it is not possible to actively query a domain expert, which model performs best among BALIF-based methods and more classical approaches such as Random Forests?
2. If a domain expert is available, is it more beneficial to allow the model to actively query specific points, or is it sufficient to label random samples? Which querying strategy is most effective for BALIF?
3. Among the active learning algorithms based on Isolation Forests, which performs best?

To address Question 1, we evaluate performance under a random query strategy, which simulates a passive weakly supervised setting where labelled information is sampled without guidance (Figure 4.6). This allows us to assess ALIF and BALIF as weakly supervised methods by removing the active learning (AL) step, and to compare them with a Random Forest trained only on labelled data. Our results show that BALIF consistently improves over vanilla IF and is generally preferable to Random Forests when labelled data are scarce. However, the benefits of the weakly supervised setting are dataset-dependent. BALIF tends to perform best when the underlying IF achieves moderately good performance that can be further refined by incorporating labelled information. Interestingly, in this passive setting, BALIF is systematically equal to or better than ALIF.

To answer Question 2, we compare passive (random query) and active query strategies within BALIF. We further investigate the impact of different query strategies (anomalous vs. margin) and ensemble prediction functions (Figures 4.7). Using the passive BALIF variant as a baseline, we find that the choice of ensemble prediction function has limited influence, with the naive variant generally performing best. In contrast, the querying strategy plays a much larger role. The margin strategy emerges as the most stable option, consistently outperforming random queries. The anomalous query strategy can be slightly superior on some datasets but occasionally underperforms random querying. Overall, active querying strategies clearly outperform passive ones.

Finally, to answer Question 3, we compare BALIF, ALIF, ν -SSVM, and IF-AAD (Figure 4.8). While performance varies across datasets, BALIF (orange) achieves the best results overall, followed closely by ALIF (blue).

Both methods substantially outperform IF-AAD (green) and ν -SSVM (red) in most cases.

4.5 Conclusion

The motivation of this work was to extend the core ideas of ALIF [BCMS22] into a rigorous Bayesian framework. This perspective not only clarifies the underlying reasons for the effectiveness of the simple heuristic rules proposed in the original paper, but also enables natural extensions of the algorithm.

The resulting Bayesian interpretation, BALIF, has proved to be a viable alternative, offering multiple dials to adapt the model to different scenarios. For instance, the relative strength of the prior can be tuned to control the influence of the underlying IF model, and the query strategy can be reformulated to align more closely with the principles of active learning, i.e., requesting information that reduces uncertainty. We also explored the trade-offs between different query strategies, illustrating qualitatively how they affect model behaviour on a toy dataset. In particular, we introduced sub-sampling as a way to mitigate the density dependence of the anomalous strategy, while retaining its benefits in prioritizing high-value queries. Moreover, BALIF directly models anomaly rates, avoiding the need to heuristically map scores into the interval $[0, 1]$. This yields predictions that are both interpretable and comparable across datasets, as they represent the models actual confidence that an observation is anomalous.

Despite these advantages, BALIF has some limitations. The model inherits the space partitions defined by the base IF, and is thus constrained by the latter's ability to capture the data domain. This may become a significant limitation in certain datasets, particularly under distribution shift, where the model cannot efficiently adapt. Extending BALIF to other tree-based tessellation methods, such as Extended Isolation Forest or Mondrian Forest, is a promising direction.

Additionally, simplifying assumptions were made to maintain tractability. While these lead to a pragmatic approximate solution, they provide limited guarantees regarding the optimality of the update procedure. Relaxing such assumptions could improve accuracy, though at the cost of increased complexity. Potential future improvements include a more detailed modelling of the likelihood function, taking into account the querying process as well as a more principled treatment of the prior derived from the baseline unsupervised model. Other possible avenues of improvement is given by the refinement of

the ensemble prediction function and consideration of intra-region interactions by tracking properties of non-terminal nodes. Finally, the developed method can be extended to other tree-based tessellation algorithms for more flexible data partitioning.

Our findings suggest that this Bayesian reformulation is a promising step toward more flexible and interpretable active anomaly detection methods.

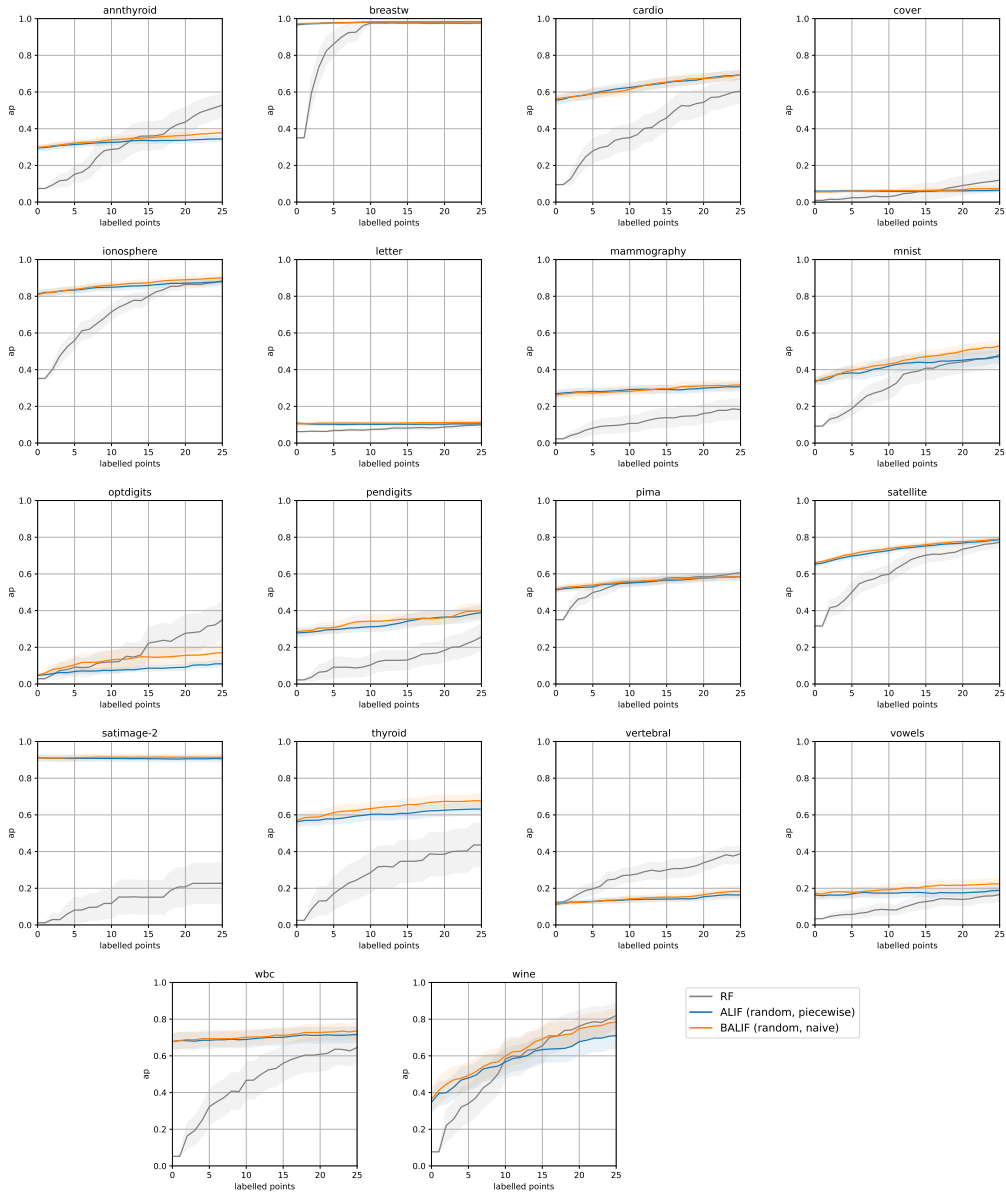


Figure 4.6: Comparison between RF, ALIF and BALIF as WS models (no active query). The figure shows how the model’s Average Precision improves as a function of the number of labelled data and is compared to the one achieved with a random forest trained on the same amount of labelled data.

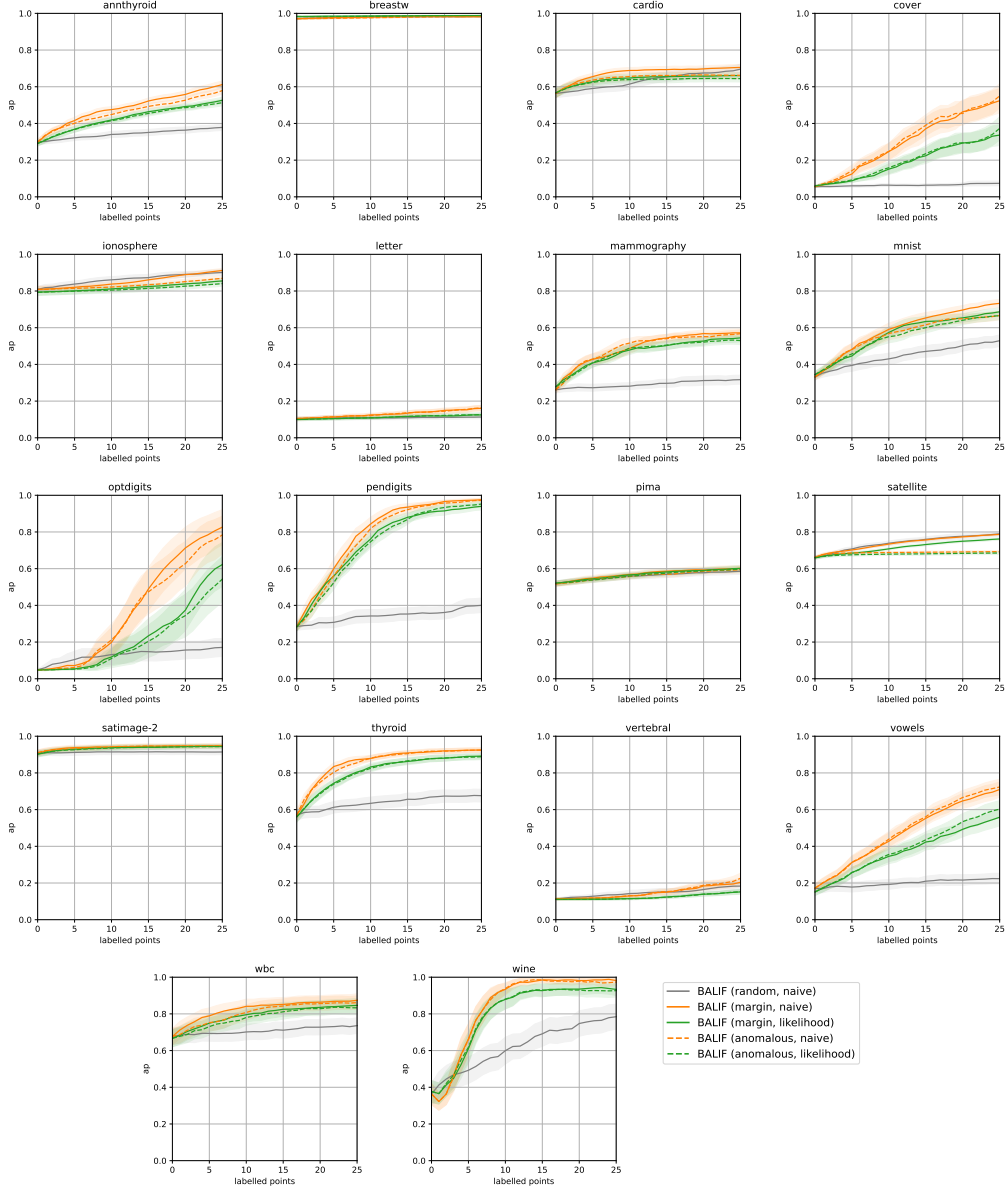


Figure 4.7: Comparison between different variants of BALIF on real world data sets. The figure shows how the model’s Average Precision improves as a function of the number of queries for all combination of ensemble predictions and querying strategies, compared with a weakly supervised variant (random query).

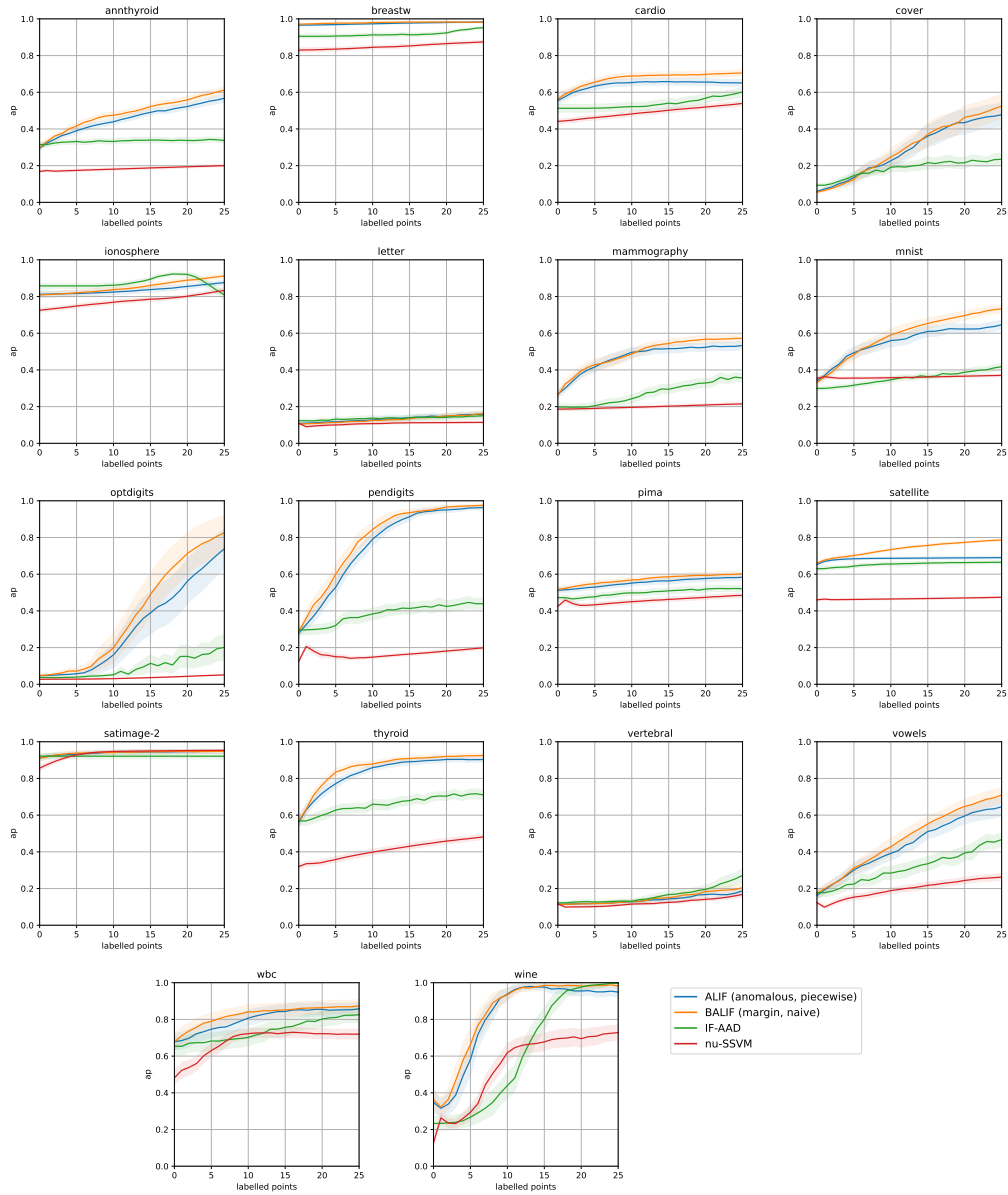


Figure 4.8: Comparison between different Active Learning algorithms. BALIF and ALIF generally outperform other methods while also introducing significantly less overhead.

5

Extended Bayesian Active Learning Isolation Forest

5.1 Introduction

Anomaly Detection (AD) is a fundamental task in engineering, concerned with identifying deviations from expected patterns in data. Data-driven methods have gained prominence in recent years, as deriving explicit models of complex systems is often infeasible. Among them, unsupervised approaches are especially attractive due to the frequent unavailability or high cost of labelled data. Their main advantage lies in not relying on predefined models, which makes them well suited for monitoring complex automated systems and processes. Such methods are often integrated into broader frameworks, such as Decision Support Systems (DSS), to assist domain experts in monitoring tasks.

While unsupervised AD is effective, incorporating limited human feedback can further improve detection performance. In this context, active learning (AL) strategies provide a principled way to select and integrate such feedback. This chapter introduces BALEIF, a Bayesian active learning framework for the Extended Isolation Forest (EIF) algorithm. BALEIF builds on BALIF

[SBS24a], which originally proposed a Bayesian extension of the Isolation Forest (IF) for iterative integration of expert feedback.

IF is one of the most popular anomaly detection algorithms due to its efficiency and strong empirical performance. However, it is known to introduce biases and artifacts in its decision function, which may limit its reliability [HKB19]. EIF was introduced in [HKB19] to mitigate these issues.

In this chapter, we extend the BALIF framework to EIF, demonstrating that it can be seamlessly integrated with the same theoretical and practical benefits. Through experiments on synthetic and real-world datasets, we validate the effectiveness of BALEIF, and through comparative analysis we show that it generally outperforms BALIF in settings where EIF and IF achieve similar baseline performance.

5.2 Extended Isolation Forest

The Isolation Forest (IF) algorithm provides a simple and effective framework for isolation-based anomaly detection. However, it suffers from fundamental flaws and biases. Over the years, several variants of IF have been proposed, either to address these shortcomings or to adapt the method to specific scenarios. In this section, we briefly introduce one such variant: the Extended Isolation Forest (EIF) [HKB19].

In the original IF, partitions are created using axis-aligned cuts. At each step, one input dimension is chosen uniformly at random, and the split is placed at a random point between the minimum and maximum values along that dimension. This procedure results in terminal regions with edges strictly aligned to the coordinate axes, leading to a pronounced block bias. As a consequence, the anomaly score function can exhibit artifacts, particularly in datasets with complex distributions [HKB19].

EIF addresses this limitation by generalizing the splitting process. Instead of restricting partitions to axis-aligned hyperplanes, EIF samples a random direction in the input space. Specifically, each split is defined by:

- a random normal vector $\hat{n}$, sampled uniformly from the unit sphere
- a random point p within the current region.

These two elements define a hyperplane that partitions the region into two subregions. To determine which side of the hyperplane a data point d falls on, one simply evaluates the sign of its projection:

$$\hat{n} \cdot d - \hat{n} \cdot p. \quad (5.1)$$

The key difference between IF and EIF lies in the direction of the splits. While IF uses axis-parallel hyperplanes, EIF samples hyperplanes with arbitrary orientations. As a result, EIF eliminates the axis-aligned bias while preserving all the desirable properties of IF, typically producing more reliable anomaly scores.

For example, Figures 5.1b and 5.1e show anomaly score heatmaps for a two-dimensional, two-cluster dataset under IF and EIF, respectively. In the IF case, axis-aligned cuts produce spurious safe regions that appear less anomalous than they should, undermining robustness. EIF removes these artifacts, ensuring that equally anomalous points are treated consistently.

This extension comes at the cost of a modest computational overhead. Since splits are no longer axis-aligned, evaluating the path of a point through the trees requires additional vector operations, whose complexity increases with the data dimensionality. Nevertheless, the gain in robustness often justifies this added cost.

5.3 BALEIF

Bayesian Active Learning Isolation Forest (BALIF) is an active learning algorithm that refines IF models within a probabilistic framework. While originally built on top of IF, its formulation makes few assumptions on the underlying anomaly detection model, relying only on the induced tree structures. We extend the approach by considering Extended Isolation Forest (EIF) as the base algorithm, introducing BALEIF as the main contribution.

We first briefly recall the main components of BALIF introduced in the previous chapter, in order to highlight of the method can be extended to a different baseline AD model.

In BALIF, each terminal region (leaf) L of an IF tree is associated with an anomaly rate $\text{Ar}(L) \in [0, 1]$, defined as the probability that a point $x \in L$ is anomalous:

$$\text{Ar}(L) \doteq \text{P}[x \text{ is anomalous} \mid x \in L]. \quad (5.2)$$

Since $\text{Ar}(L)$ is unknown, BALIF maintains a belief distribution $g_L(z) \doteq \hat{\text{P}}[\text{Ar}(L) = z]$. Each belief is modeled as a Beta distribution,

$$g_L(z) \sim \text{Beta}(\alpha_L, \beta_L) \propto z^{\alpha_L-1}(1-z)^{\beta_L-1}, \quad (5.3)$$

with priors initialized from the anomaly scores of the underlying unsupervised model, treating each tree as a weak predictor. Concretely, a flat prior

is minimally perturbed so that its mean matches the prediction of the corresponding tree.

The core of BALIF is its model update algorithm based on Bayesian inference, which allows for efficient integration of feedback (in the form of labelled data) into the model in an online fashion. As new labeled data points $(x, y) \in \mathbb{R}^d \times \{0, 1\}$ are available to the model, assuming independence of $\text{Ar}(L)$ for non-overlapping regions and for partially overlapping regions in different trees, the belief of each leaf is updated using Bayes rule.

$$g_L(z|x \in L, y) = \frac{p(x \in L, y|z)g_L(z)}{p(x \in L, y)}. \quad (5.4)$$

The likelihood term factorizes as:

$$\begin{aligned} p(x \in L, y | z) &= p(x \in L | z), p(y | x \in L, z) \\ &= p(x \in L | z), z^y(1 - z)^{1-y}. \end{aligned} \quad (5.5)$$

If we further assume that the sampling of x is independent of its anomaly probability, the likelihood reduces to a Bernoulli distribution which makes the Bayes update analytically tractable, requiring only an increment of the parameters in the terminal regions where x falls:

$$\begin{aligned} g_L(z | x \in L, y) &\propto z^{\alpha-1}(1 - z)^{\beta-1}z^y(1 - z)^{1-y} \\ &\sim \text{Beta}(\alpha + y, \beta + 1 - y). \end{aligned} \quad (5.6)$$

This procedure enables BALIF to iteratively refine its predictions using feedback, while introducing no significant computational overhead. However, the update step does not alter the underlying partitions of the input space, which remain those induced by the base IF. As a result, any artifacts or biases in the partitioning persist even after incorporating labels, and may propagate into the updated anomaly scores.

Importantly, though BALIF inherits IFs space partitions, its update process is not tied to IFs specific structure. In fact, none of the steps described so far are specific to the IF structure and, in principle, the framework applies to any ensemble method that partitions the input space. Building on this observation, we can introduce BALEIF, which replaces IF with EIF, thereby mitigating the axis-aligned biases and artifacts of the original IF while retaining the benefits of the Bayesian update mechanism.

Finally, the ensemble model prediction can also be obtained following the same procedure as BALIF. For a given data point $x \in \mathbb{R}^d$, the prediction of

an individual tree T is given by the distribution $g_{L_{x,T}}(z)$, where $L_{x,T}$ is the terminal region of T containing x . To obtain a global anomaly score, predictions from multiple trees are aggregated. Among several possible strategies, the most effective in practice is the naive arithmetic mean across trees. Taking the mean of each leafs Beta distribution, the anomaly score is defined as:

$$a(x) = \frac{1}{n_T} \sum_T \frac{\alpha_{L_{x,T}}}{\alpha_{L_{x,T}} + \beta_{L_{x,T}}}, \quad (5.7)$$

where n_T is the number of trees.

The original derivation of this formula followed a more principled probabilistic argument, in which the global prediction itself was modeled as a probability distribution. We refer the reader to [SBS24a] for a detailed discussion of these alternatives. This simple and principled choice worked out of the box also for BALEIF, and we did not investigate the effects of alternative ensemble prediction strategies when using EIF for the space partitions.

5.4 Time complexity

The time complexity of IF, BALIF, EIF and BALEIF can be found in Table 5.1. Comparing IF and BALIF to EIF and BALEIF, since the latter involve operations on d dimensional hyper-planes, the time complexity technically increases by a factor of d . However, in practice, this increase is hardly noticeable. The overhead introduced by the active models is minimal and is dominated by the tree operations of the base model. For the fit operation, only a couple additional multiplications are needed, which is also true for inference. Similarly, the update procedure is dominated by the tree traversal and has the same complexity as inference.

5.5 Experimental Results

In this section, we present experimental results on both synthetic and real-world datasets. We compare BALEIF with BALIF, showing that in several scenarios BALEIF achieves superior performance. In these experiments, we adopt a more efficient variant of EIF, namely the Generalized Isolation Forest proposed in [LBST21a], instead of the original EIF implementation. The GIF algorithm refines the hyperplane sampling procedure of EIF and prevents the creation of empty branches, thereby improving both efficiency and robustness.

	Fit	Prediction	Update
IF	$O(n_T \cdot \psi)$	$O(n_T \cdot \log \psi)$	
BALIF	$O(n_T \cdot \psi)$	$O(n_T \cdot \log \psi)$	$O(n_T \cdot \log \psi)$
EIF	$O(n_T \cdot d \cdot \psi)$	$O(n_T \cdot d \cdot \log \psi)$	
BALEIF	$O(n_T \cdot d \cdot \psi)$	$O(n_T \cdot d \cdot \log \psi)$	$O(n_T \cdot d \cdot \log \psi)$

Table 5.1: Time complexities for the considered approaches as a function of the number of estimators in the forest n_T and the sub-sample size used for bagging ψ , which determines the maximum tree depth $\log \psi$.

5.5.1 Synthetic datasets

In this section, we evaluate the performance of BALEIF on two synthetic two-dimensional datasets, illustrated in Fig. ???. These datasets, adapted from those used in the original EIF paper [HKB19], are deliberately designed to be challenging for IF. They also highlight the advantages of semi-supervised approaches: anomalous labels are assigned in a way that conflicts with the inductive biases of both IF and EIF. As a result, these datasets are difficult not only for IF but also for unsupervised EIF.

The first dataset consists of two clusters, where the smaller cluster in the bottom-right corner is defined as anomalous. Figures 5.1b and 5.1e show the anomaly score heatmaps of IF and EIF without any labels. The yellow regions indicate that neither model identifies the anomalous cluster correctly. This failure reflects the assumption underlying both methods: anomalies are few and different. In this case, however, anomalies are clustered and therefore similar to each other, violating the second assumption. Figures 5.1c and 5.1f show the heatmaps after incorporating a single label. The labelled point is selected using the active learning strategy of [SBS24a], described in Chapter 4. With only one feedback label, both BALIF and BALEIF show substantial improvements. After five labels (Figures 5.1c and 5.1g), the anomalous cluster is correctly identified. However, BALIF still exhibits the characteristic artifacts of IF, whereas BALEIF eliminates these distortions, resulting in a cleaner and more accurate anomaly score map.

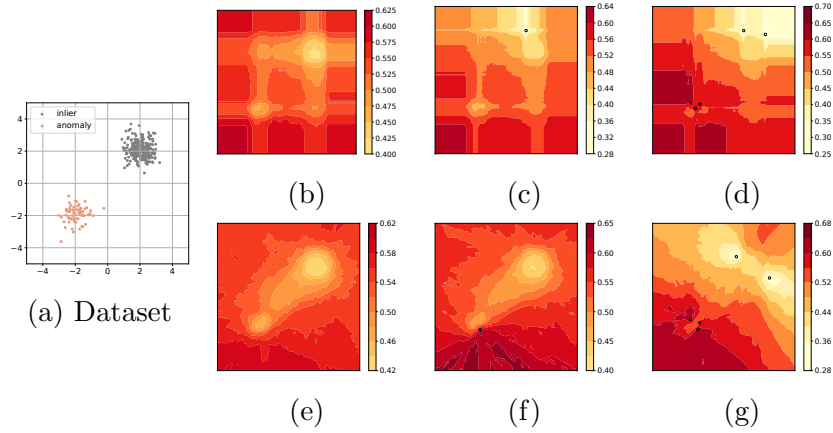


Figure 5.1: Experiments on *double-cluster* synthetic dataset. Panel (a) shows the data distribution. The first column shows the initial model predictions, (b) for IF and (e) for EIF. The second column shows the predictions after 1 query, (c) for BALIF, (f) for BALEIF, while the third column shows the predictions after 5 queries, (d) for BALIF, (g) for BALEIF.

The second dataset follows a sinusoidal pattern (Fig. 5.2a). It is characterized by sparser points in the left and right tails, making those regions easier to isolate for both IF and EIF. The challenge arises from labeling only the points on the right tail as anomalous. As shown in Figs. 5.2b and 5.2e, both IF and EIF mistakenly assign high anomaly scores to points in the left tail, which are in fact normal, leading to a high rate of false positives. After incorporating a single label, the anomaly score maps of BALIF and BALEIF (Figs. 5.2c and 5.2f) already reflect differences in how the two frameworks partition the input space. Owing to the randomness in tree construction, query points are drawn from different regions; nevertheless, the structural differences in space partitioning clearly influence the models evolution. With five labels, BALEIF successfully distinguishes the anomalous right tail from the normal left tail, yielding a consistent separation of the two regions. By contrast, BALIF fails to make this distinction, continuing to misclassify left-tail points as anomalous. This highlights BALEIFs ability to overcome the artifacts of IF while requiring only a small amount of labelled feedback.

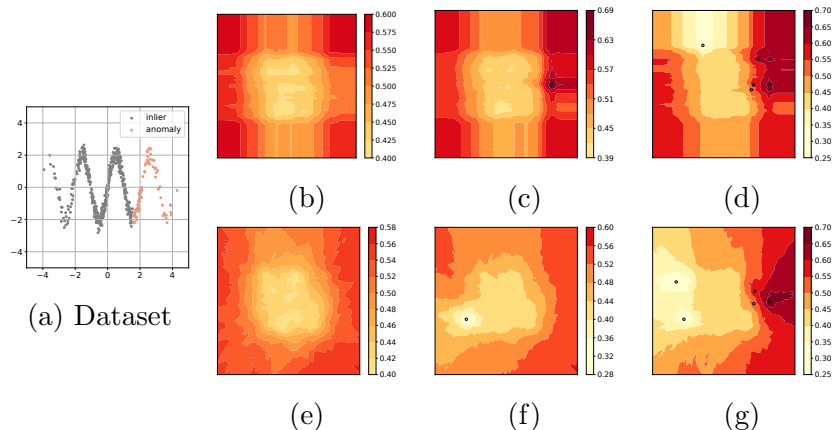


Figure 5.2: Experiments on *sinusodal* synthetic dataset. Panel (a) shows the data distribution. The first column shows the initial model predictions, (b) for IF and (e) for EIF. The second column shows the predictions after 1 query, (c) for BALIF, (f) for BALEIF, while the third column shows the predictions after 5 queries, (d) for BALIF, (g) for BALEIF.

5.5.2 Real-world dataset

Following the experimental setup of BALIF, we evaluate BALEIF on 17 real-world datasets from the Outlier Detection Datasets (ODDS) repository [Ray16]. These datasets are derived from multi-class classification problems, where the smallest classes are treated as outliers and the remaining ones as inliers. Because of their original design for classification tasks, outliers are not arbitrary but concentrated in specific regions of the feature space and exhibit structured patterns. This once again violates the few and different assumption underpinning both IF and EIF. We train and test our models on the entire datasets.

For evaluation, visualizations of anomaly score maps are impractical due to the high dimensionality of the data. Instead, we follow BALIF and adopt the average precision score (AP) as the main performance metric. Anomaly detection can be cast as a binary classification task: models assign anomaly scores to data points, which are then thresholded to predict normal or anomalous instances. Since anomalies are rare, this results in a highly imbalanced classification problem, where precision and recall are more meaningful than accuracy. To avoid dependence on a specific threshold, we use the AP score, which summarizes precision-recall performance across thresholds.

In our experiments, we allow each model to perform a maximum number of queries equal to 10% of the dataset size. We then plot the AP score as a function of the number of queries (Fig. 5.3). The results show that, while neither method consistently dominates across all datasets, BALEIF achieves higher average precision than BALIF on a non-negligible subset of them, demonstrating its potential advantages in real-world settings.

5.6 Conclusion

We presented BALEIF, a modification of BALIF that replaces IF with EIF, while retaining the same Bayesian foundations and computational efficiency. The motivation behind this work was to address the well-documented artifacts and biases introduced by IF, which also propagate to BALIF.

Through synthetic experiments, we demonstrated that BALEIF effectively removes these artifacts while preserving the ability to integrate feedback and improve the alignment of anomaly scores with end-user expectations. On real-world datasets, BALEIF shows competitive performance and outperforms BALIF in a non-negligible number of cases, particularly in data distributions where EIF provides a better unsupervised characterization of the domain.

Looking ahead, future improvements could involve relaxing the tight coupling between the update procedure and the partitioning induced by the underlying tree structure. In particular, modelling intra-region interactions could allow the framework to better capture complex dependencies across regions, further enhancing robustness and adaptability.

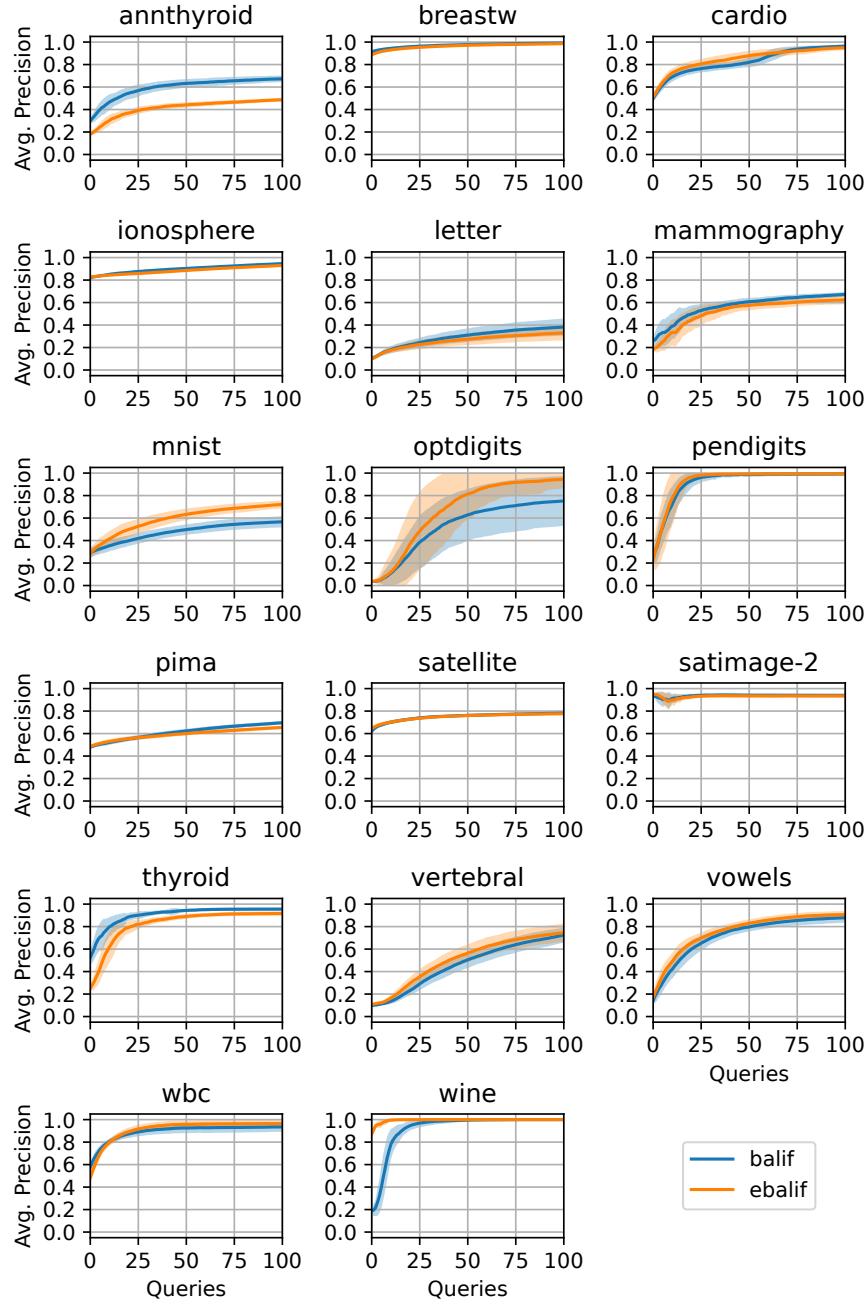


Figure 5.3: Comparison between BALEIF and BALIF on real world dataset.

6

Batch Bayesian Active Learning Isolation Forest

6.1 Introduction

In active learning (AL), it is often desirable to query labels in batches rather than sequentially, both to optimize labeling time and to reduce the burden on domain experts. However, most query strategies are designed for single-point selection, and when applied naively to batch settings they often fail due to data redundancy. This chapter proposes a new batch-querying strategy for the Bayesian Active Learning Isolation Forest (BALIF), which was originally designed for sequential, single-point queries. We compare our approach with two established Bayesian querying strategies (BALD [HHGL11] and its batch extension BatchBALD [KvAG19]) adapted here to the BALIF framework.

Experiments on multiple anomaly detection benchmark datasets with data duplication show that our method achieves superior performance. By selecting informative samples in batches, our approach enables domain experts to provide feedback in fewer interactions, making their involvement more efficient and practical.

Two existing AL extensions of the Isolation Forest (IF) are the Active

Learning Isolation Forest (ALIF) [MBSS24] and BALIF [SBS24b]. Both query points sequentially, one at a time. While effective, this setup requires frequent user interactions, which can be inefficient and inconvenient given the cost of expert involvement [SPY+24]. From a usability perspective, it is preferable for the model to propose a batch of informative points jointly, reducing the number of expert interactions.

A naive batch strategy would be to select the top- k most informative points. However, this approach has two drawbacks. First, it ignores correlations between queries: labeling one point may reduce or even eliminate the informativeness of others. Second, in datasets with duplicates or near-duplicates, such a strategy risks wasting queries on redundant points, thereby diminishing the value of the batch.

Probabilistic information-gain methods provide a natural solution. BALD was originally introduced for Gaussian Process classifiers and later extended to batch settings in deep learning as BatchBALD. Their theoretical grounding makes them attractive candidates for BALIF, particularly in addressing the batch-query problem.

In this work, we explore and compare batch querying strategies for BALIF. We derive a closed-form formulation of BALD and BatchBALD within the BALIF framework and contrast them with a new heuristic-based method that leverages BALIF’s native interest scores. Our experiments show that the proposed robust worst-case method achieves the best overall performance, while remaining more flexible and computationally efficient than existing alternatives.

6.2 Preliminaries

The Bayesian Active Learning Isolation Forest (BALIF) [SBS24b] is a probabilistic anomaly detection algorithm that extends the popular Isolation Forest (IF) with an active learning strategy. In this section, we present BALIF through a Bayesian lens, which provides the foundation for applying principled Bayesian active learning methods.

Although IF is inherently unsupervised, recent extensions have introduced active learning capabilities. The Active Learning Isolation Forest (ALIF) [MBSS24] is a weakly supervised variant that updates leaf-node depths, while BALIF generalizes ALIF by defining a rigorous Bayesian framework for model updates.

Each decision tree in the IF ensemble induces a partition of the feature

space, represented by its leaves. This partitioning remains fixed in ALIF and BALIF; only the associated parameters are updated. Specifically, BALIF introduces a parameter θ_l for each leaf l , representing the probability that a point $x \in l$ is anomalous:

$$y \mid \theta \sim \text{Bernoulli}(\theta_l). \quad (6.1)$$

The belief over θ_l is modeled as a Beta distribution:

$$\theta_l \sim \text{Beta}(\alpha_l, \beta_l). \quad (6.2)$$

The parameters (α_l, β_l) are initialized according to the depth-based anomaly score of IF and subsequently updated with labeled feedback. Given a labeled sample x with true label $\hat{y}$, the posterior for θ_l is updated for all leaves l containing x :

$$\theta_l \mid \hat{y} \sim \text{Beta}(\alpha_l + \hat{y}, \beta_l + 1 - \hat{y}). \quad (6.3)$$

For prediction, BALIF aggregates the contributions of all leaves containing a given point x . The ensemble predictive distribution G_x is itself a Beta distribution:

$$y \sim G_x = \text{Beta}(\nu_x \mu_x, \nu_x (1 - \mu_x)), \quad (6.4)$$

where $\mu_x = \mathbb{E}_{l \ni x} \left[\frac{\alpha_l}{\alpha_l + \beta_l} \right]$ and $\nu_x = \sum_{l \ni x} (\alpha_l + \beta_l)$. Here, μ_x is the mean anomaly rate and ν_x acts as an effective sample size.

To select queries, [SBS24b] propose two heuristic interest scores based on $G_x(z)$. The anomalous strategy queries points with the highest anomaly score, i.e., the mean of G_x :

$$\text{IS}_{\text{ANOM}}(x) = \mathbb{E}_{z \sim G_x} [z] = \mu_x. \quad (6.5)$$

The likelihood margin strategy is specific to BALIF and it prioritizes points near the decision boundary. For a threshold r , the score is the negative likelihood ratio between the mode of G_x and its value at r :

$$\text{IS}_{\text{MARGIN}}(x) = -\frac{\max_z G_x(z)}{G_x(r)}. \quad (6.6)$$

6.3 BALD

Bayesian Active Learning by Disagreement (BALD) [HHGL11] is an active learning approach that selects data points based on the mutual information

between the predicted label y and the model parameters θ . The core intuition is that the most informative queries are those where predictions are strongly coupled to model parameters: labeling such points yields the largest refinement of the posterior.

Let θ denote the model parameters with posterior distribution $p(\theta)$. For an unlabeled sample x , BALD computes the conditional mutual information between the posterior predictive distribution over y and the posterior over θ :

$$\begin{aligned} \text{IS}_{\text{BALD}}(x) &= I(y ; \theta) \\ &= H(y) - \mathbb{E}_{\theta \sim p(\theta)} [H(y | \theta)], \end{aligned} \quad (6.7)$$

where $H(\cdot)$ denotes Shannon entropy.

The first term, $H(y)$, measures the overall uncertainty in the model's prediction for x . A high value indicates that the model, marginalizing over parameter uncertainty, is unsure about the label. The second term, $\mathbb{E}_{\theta} [H(y | \theta)]$, is the expected predictive entropy of individual models sampled from the posterior. A low value here indicates that each sampled model makes a confident prediction.

Thus, BALD scores are high when the model is globally uncertain about x (large prediction entropy $H(y)$), but individual models disagree with one another (small expected entropy). This situation occurs only when different posterior samples are confident yet contradictory in their predictions. Such points are maximally informative: their labels resolve disagreements among plausible models, yielding the greatest reduction in posterior uncertainty.

6.4 Closed-form BALD in BALIF

BALD is a Bayesian acquisition strategy that requires access to the posterior distribution of the model parameters. This makes it particularly well-suited for the BALIF framework, where parameter uncertainty is explicitly modeled via Beta distributions representing the anomaly rates of the leaves in each tree. Thanks to the closed-form nature of these Beta distributions, we can derive an exact expression for the BALD interest scores from eq. (6.7), avoiding the need for Monte Carlo sampling as in [KvAG19].

For a single tree, the posterior predictive distribution of a leaf l is given directly from eq. (6.1):

$$y \sim \text{Bernoulli} \left(\frac{\alpha_l}{\alpha_l + \beta_l} \right). \quad (6.8)$$

The entropy of this predictive distribution is therefore:

$$H(y) = -\frac{\alpha_l}{\alpha_l + \beta_l} \log \left(\frac{\alpha_l}{\alpha_l + \beta_l} \right) - \frac{\beta_l}{\alpha_l + \beta_l} \log \left(\frac{\beta_l}{\alpha_l + \beta_l} \right). \quad (6.9)$$

The conditional entropy term, corresponding to the expected predictive entropy under the Beta posterior, also admits a closed-form expression in terms of the digamma function ψ :

$$\begin{aligned} \mathbb{E}_{\theta \sim p(\theta)}[H(y \mid \theta)] &= \mathbb{E}_{\theta_l \sim \text{Beta}(\alpha_l, \beta_l)}[H(y \mid \theta_l)] \\ &= \psi(\alpha_l + \beta_l + 1) - \frac{\alpha_l \psi(\alpha_l + 1) + \beta_l \psi(\beta_l + 1)}{\alpha_l + \beta_l}. \end{aligned} \quad (6.10)$$

Substituting eqs. (6.9) and (6.10) into eq. (6.7) yields the exact BALD interest score for BALIF leaves, making the acquisition strategy both efficient and numerically stable.

6.5 BatchBALD

In practice, top- k BALD batch selection is implemented by choosing the k points with the highest individual mutual information scores, i.e., those maximizing:

$$\text{IS}_{\text{BALD top-}k}(x_1, \dots, x_k) = \sum_{i=1}^k I(y_i; \cdot, \theta). \quad (6.11)$$

However, this strategy implicitly assumes that each point contributes information independently. In reality, selected points may be redundant or highly similar, which reduces the overall informativeness of the batch.

BatchBALD [KvAG19] addresses this limitation by evaluating the informativeness of a set of points jointly, rather than treating them independently. Instead of summing mutual information scores, it computes the mutual information between the set of predicted labels and the model parameters:

$$\text{IS}_{\text{BatchBALD}}(x_1, \dots, x_k) = I(y_1, \dots, y_k; \cdot, \theta). \quad (6.12)$$

By capturing correlations among points, BatchBALD avoids redundancy and encourages selection of more diverse, complementary samples. This property makes it particularly effective when datasets contain duplicated or near-duplicate points.

A direct computation of the joint mutual information for all possible batches is intractable, as complexity grows rapidly with batch size. To make

the method practical, BatchBALD uses a greedy approximation: points are added to the batch one at a time, each chosen to maximize its conditional mutual information given the already-selected points. This approach avoids exhaustive evaluation of all subsets, while providing a close approximation to the optimal batch selection, as demonstrated in [KvAG19].

6.6 Closed-form BatchBALD in BALIF

BatchBALD in BALIF. BatchBALD can be adapted to BALIF in a similar way as BALD, but requires consideration of joint entropies and joint conditional entropies.

For a single tree in the BALIF ensemble and a batch of points with predictions $y_1, \dots, y_k$, predictions corresponding to different leaves are independent. Thus, the joint entropy decomposes into a sum of leaf-level entropies:

$$H(y_1, \dots, y_k) = \sum_l H(y_{l_1}, \dots, y_{l_{n_l}}). \quad (6.13)$$

For points within the same leaf l , the posterior predictive distribution is:

$$\begin{aligned} \mathbb{P}(y_{l_1}, \dots, y_{l_{n_l}}) &= \mathbb{E}_\theta [\mathbb{P}(y_{l_1}, \dots, y_{l_{n_l}} \mid \theta)] \\ &= \frac{B(\alpha_l + m_l, \beta_l + n_l - m_l)}{B(\alpha_l, \beta_l)} = \frac{\alpha_l^{(m_l)} \beta_l^{(n_l - m_l)}}{(\alpha_l + \beta_l)^{(n_l)}} \end{aligned} \quad (6.14)$$

where $x^{(n)} = \prod_{i=0}^{n-1} (x + i)$ is the rising factorial, $m_l = \sum_{i=1}^{n_l} y_i$ is the number of anomalies, and $B(\cdot, \cdot)$ is the Beta function.

Grouping outcomes with the same m_l , the entropy of the joint leaf predictions can be expressed as:

$$H(y_{l_1}, \dots, y_{l_{n_l}}) = - \sum_{m_l=0}^{n_l} \binom{n_l}{m_l} \frac{\alpha_l^{(m_l)} \beta_l^{(n_l - m_l)}}{(\alpha_l + \beta_l)^{(n_l)}} \log \left(\frac{\alpha_l^{(m_l)} \beta_l^{(n_l - m_l)}}{(\alpha_l + \beta_l)^{(n_l)}} \right). \quad (6.15)$$

The conditional entropy is simpler to compute. Since y_i are conditionally independent given θ , the joint conditional entropy factorizes as:

$$\mathbb{E}_{\theta \sim p(\theta)} [H(y_1, \dots, y_k \mid \theta)] = \sum_{i=1}^k \mathbb{E}_{\theta \sim p(\theta)} [H(y_i \mid \theta)], \quad (6.16)$$

where each term can be computed in closed form as in eq. (6.10).

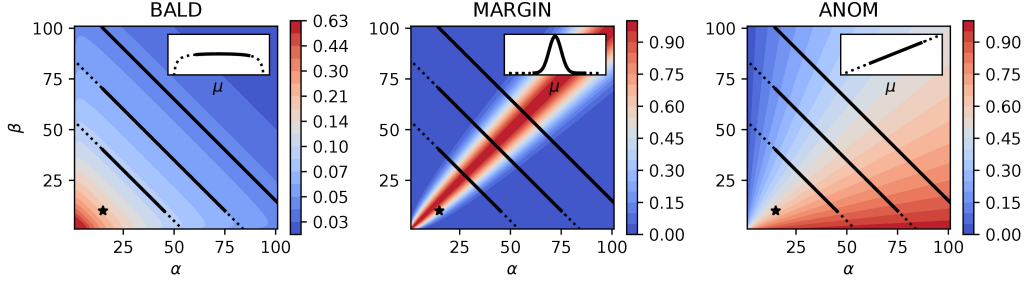


Figure 6.1: Interest scores as functions of α and β under different querying strategies. The star indicates the initial state before any update. Black lines represent contours of constant sample size $\nu = \alpha + \beta$ after updates. Since each label update only increments α or β , the parameters can only move along the solid portion of the line, corresponding to achievable states. Along any such line, the interest scores are unimodal, which implies that the minimum of the score lies at one of the two extrema of the reachable interval.

Finally, as in the original BALIF framework, the mutual information over the entire ensemble is obtained by summing across trees, under the assumption of independence. In practice, however, computing the exact mutual information for all possible batches remains intractable. Therefore, as in BatchBALD [KvAG19], we adopt a greedy approximation, selecting points sequentially based on their incremental contribution to the mutual information given the points already chosen (see section 6.5).

6.7 Robust-Worst-Case Batch Querying

The approach of [KvAG19] for extending BALD to batch queries cannot be directly generalized to other acquisition strategies. To address this, we propose the robust-worst-case (RWC) strategy, which provides a general way to adapt any sequential interest score to a batch setting. RWC follows the same greedy procedure as BatchBALD (see section 6.5), but adopts a conservative criterion when selecting each new sample.

The procedure starts by selecting the first candidate point as in the sequential case:

$$x_1 = \arg \max_x \text{IS}(x).$$

The challenge arises when selecting subsequent samples: the optimal choice depends on the unknown labels $\hat{y}_1, \dots, \hat{y}_j$ of the already selected points.

To handle this, RWC simulates all possible outcomes for these labels and assigns the next candidate an interest score equal to the minimum score across scenarios. Formally, for a partial batch $x_1, \dots, x_j$, the score of a candidate x is:

$$\text{IS}_{\text{RWC}}(x; x_1, \dots, x_j) = \min_{\hat{y}_1, \dots, \hat{y}_j} \text{IS}(x \mid \hat{y}_1, \dots, \hat{y}_j). \quad (6.17)$$

Naively, evaluating eq. (6.17) requires considering 2^j possible label assignments, leading to exponential complexity in the batch size. However, for most interest scores based on the predictive distribution G_x (see eq. (6.4)), the computation simplifies dramatically. Substituting the update rule from eq. (6.3) into eq. (6.17), the updated sample size is:

$$\nu_x = \sum_{l \ni x} \left(\alpha_l + \beta_l + \sum_{i=1}^j \mathbb{1}_{x_i \in l} \right), \quad (6.18)$$

which depends only on the number of queried points falling in each leaf, and not on the actual label values. Thus, all candidate distributions differ only in their mean μ_x but share the same sample size ν_x .

If the chosen interest score is unimodal along lines of constant sample size, then the minimum value (the worst case) must occur at one of the two extreme scenarios: all labels anomalous or all labels normal. Consequently, only these two cases need to be evaluated, reducing the complexity from exponential to constant in the batch size.

As illustrated in fig. 6.1, the BALD interest score (eq. (6.7)) is unimodal for fixed sample size. Similarly, the two strategies proposed in [SBS24b] (most anomalous and likelihood margin) also satisfy this property. Therefore, all three can be efficiently adapted to batch querying under the RWC framework.

6.8 Experiments

We evaluate the proposed batch-querying strategies on the well-established anomaly detection datasets from [She16b]. Each experiment uses a 50-50 stratified split into training and test sets: active learning is performed on the training set, while performance is evaluated on the test set using average precision (AP).

Since the batch-querying problem is most relevant when datasets contain real or near-duplicates, we follow [KvAG19] and duplicate the training

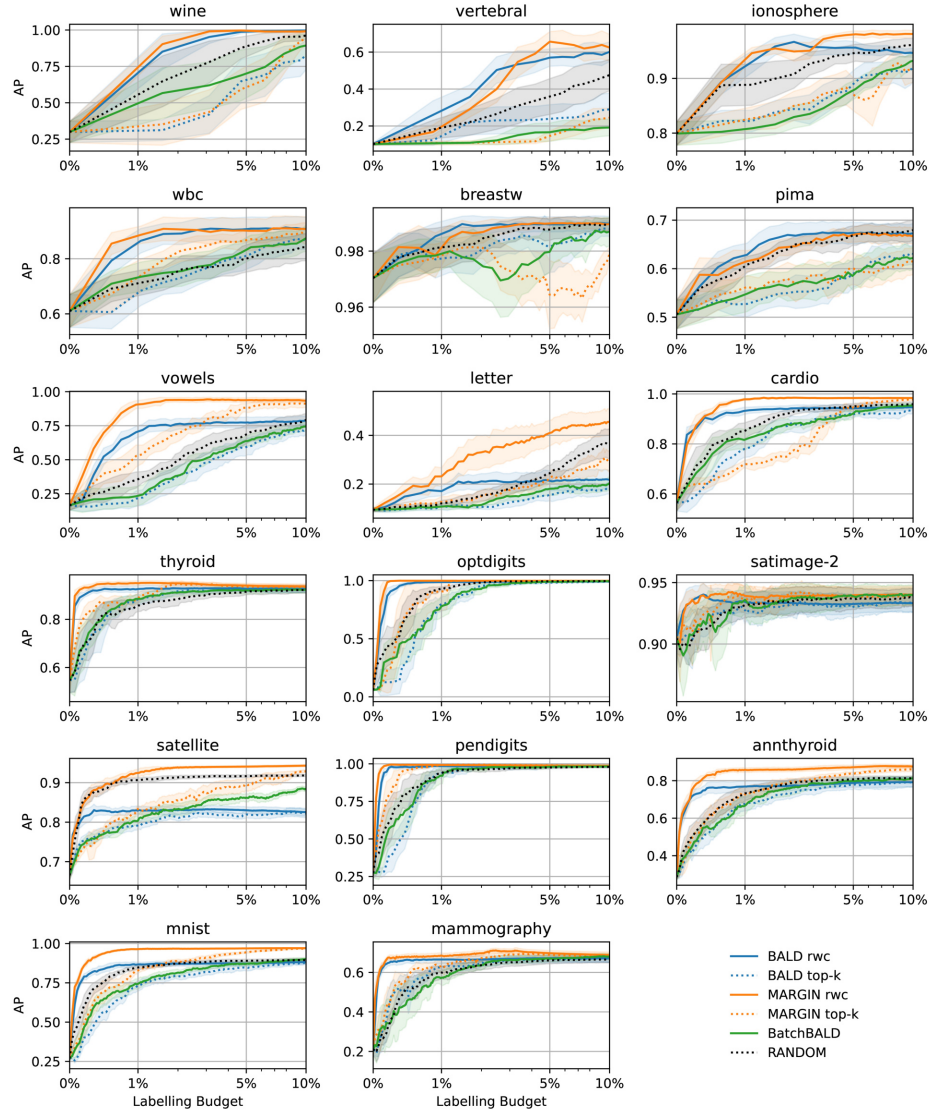


Figure 6.2: Comparison of average precision for different batch-querying strategies across datasets, ordered by increasing dataset size. The shaded regions indicate the standard deviation across repeated runs. The proposed RWC strategy consistently outperforms all alternatives. By contrast, BatchBALD underperforms even random querying across most datasets. Among the remaining strategies, the MARGIN interest score demonstrates better scalability than BALD on medium- to large-scale datasets.

data. Both the number of duplicates and the batch size are set to 10, and active learning is run until 10% of the training set has been labeled. Each experiment is repeated 10 times with different random seeds and standard hyperparameters to estimate uncertainty.

We report results in terms of AP as a function of the number of labeled points. AP is well suited for anomaly detection tasks, as it captures the trade-off between precision and recall in imbalanced datasets, unlike metrics based on false positive or false negative rates. The code to reproduce all experiments and figures is publicly available at: github.com/AMCO-UniPD/balif.

The results in fig. 6.2 show that RWC strategies consistently outperform their top- k counterparts across all datasets, validating RWC as a robust and effective approach for batch querying. Among the RWC strategies, the MARGIN method generally performs best, often surpassing BALD, with particularly strong gains on the **letter** and **vowel** datasets. For **breastw** and **satimage-2**, all methods achieve comparable performance, likely due to the already high accuracy of the unsupervised IF baseline, which leaves little room for improvement. Interestingly, on **pima**, random querying matches the performance of RWC strategies, despite IFs modest baseline performance.

Dataset size appears to play an important role. BALD RWC often plateaus after a few queries on medium and large datasets (from **vowel** onwards), indicating limited scalability. By contrast, MARGIN RWC is consistently among the top performers on small datasets and clearly outperforms all alternatives on larger datasets. This suggests that MARGIN RWC scales more effectively than BALD RWC, which shows no clear advantage even on small datasets.

Although both RWC and BatchBALD were designed for batch-query settings with duplicates, BatchBALD performs no better than random querying on all datasets, and often worse. This aligns with prior observations that BatchBALD struggles on imbalanced datasets [KvAG19]. An additional contributing factor may be the closed-form adaptation used here, which evaluates trees independently rather than the forest as a whole.

Overall, the MARGIN RWC strategy emerges as the most effective approach, combining high performance, scalability across dataset sizes, and computational efficiency.

6.9 Conclusion

Motivated by the practical need in active learning (AL) to optimize expert time through joint labeling of multiple points, we addressed the challenge of batch querying in BALIF [SBS24b]. To this end, we introduced the robust-worst-case (RWC) strategy, a novel and efficient method for batch querying. RWC relies on a conservative estimate of the interest score, selecting points that remain informative under the worst possible labeling outcomes. This ensures that each query in the batch is valuable, regardless of the unknown labels of the other queried points.

Our experiments demonstrate the effectiveness of RWC, particularly in datasets containing duplicates. The scalability of RWC on larger datasets depends on the choice of interest score: when combined with the margin score from BALIF [SBS24b], RWC achieves consistently strong performance, highlighting its practical applicability and potential to enhance user acceptance in real-world anomaly detection tasks.

Future work should investigate the role of batch size, data redundancy, and broader dataset characteristics in shaping the applicability of RWC. Further theoretical analysis of interest score properties, together with validation on large-scale and real-world benchmarks, will be essential to fully establish RWC as a reliable and widely deployable batch-querying strategy.

Part II

Safe and Interpretable Reinforcement Learning

7

Fault Identification with Safe Reinforcement Learning

7.1 Introduction

This chapter introduces a novel approach to active fault diagnosis (AFD) by explicitly separating the task into two components: passive fault diagnosis (PFD) and control input design. This formulation is highly general, and most existing AFD methods can be interpreted within this framework. By recognizing this separation, PFD techniques can be exploited to extract as much information as possible from available data, while the control input can be tailored to actively improve information gathering.

The core contribution of this work is FIERL, a general simulation-based framework for designing such control strategies. FIERL employs control-oriented reinforcement learning (CRL) to optimize the performance of arbitrary passive detectors. Importantly, the control policy can be learned without any knowledge of the internal workings of the passive detector, making FIERL broadly applicable. While it is especially effective when paired with a well-designed passive component, FIERL also extends the applicability of AFD methods to more complex scenarios, including continuous sets

of fault modes, a setting rarely addressed in the literature. Its effectiveness is demonstrated on a benchmark actuator fault diagnosis problem, where FIERL shows robust performance and the ability to generalize to fault dynamics not encountered during training.

Faults in automated systems may arise for numerous reasons (mechanical failures, design defects, user-related errors, or environmental factors) often leading to safety hazards and process disruptions. This underscores the need for reliable fault detection and isolation (FDI) modules. Early fault detection enables timely recovery strategies, preventing fault propagation and avoiding severe safety, economic, or environmental consequences. However, with the rising complexity of engineered systems and the proliferation of sensors and actuators, FDI has become increasingly challenging. While many procedures have been developed traditionally model or signal-based [Din13] recent years have witnessed growing interest in data-driven approaches. Fault Detection Identification (FDI) methods can be broadly categorized into two groups: passive fault diagnosis (PFD) and active fault diagnosis (AFD), depending on whether interactions with the system are allowed.

In PFD, process data are collected and analyzed without interfering with the system, and anomalies are identified by comparing observations against nominal input-output relations, either model-based or data-driven. Model-based PFD has been extensively studied [Din13], with well-established approaches such as residual- and observer-based strategies [PC97]. More recently, data-driven methods have also been proposed, achieving promising results in various domains [MNCHH20, GJDOP⁺21], for example by applying principal component analysis or k-nearest neighbors to system outputs.

Despite their success, PFD methods can perform poorly in realistic settings. Noise and uncertainties may obscure the subtle signatures of incipient faults, while closed-loop feedback controllers can mask fault-induced deviations. As a result, PFD may fail to detect or isolate certain fault modes reliably.

To address these limitations, AFD introduces a minimally intrusive auxiliary input in addition to the primary control signal, with the goal of generating more informative data for diagnosis. While such probing can significantly improve diagnostic accuracy, it may also degrade control performance [FK17]. This trade-off is better managed by active fault diagnosis with control (AFDC), where the primary control and auxiliary inputs are designed jointly, albeit at the cost of substantially higher complexity.

Both Active Fault Detection (AFD) and Active Fault Detection and Con-

trol (AFDC) techniques typically assume a finite set of fault modes in order to keep the problem tractable. Under this assumption, the objective is to achieve separation between different fault dynamics. This can be guaranteed theoretically by assuming deterministic and bounded disturbances [MR17, Xu21, TOSX21]. More realistic formulations, however, account for stochastic disturbances and adopt probabilistic objectives, such as minimizing the Bayesian risk of misdiagnosis [PMCM17, P15] or maximizing statistical distances between the predicted output distributions of different fault modes [PHBM18, HM17, XOP⁺17, TXWL22].

Such formulations often lead to nested optimization problems, which are difficult to solve without heavy simplifications or restrictive assumptions, particularly in the case of AFDC. To overcome these limitations, recent works have explored the use of reinforcement learning (RL) to address these optimization tasks, showing promising results [JH21, KP22, YXT⁺23, PL16].

In this work, we introduce Fault Identification Enhancement with Reinforcement Learning (FIERL), a general RL-based framework for active fault diagnosis. FIERL reformulates the AFD/AFDC paradigm by explicitly disentangling two key tasks:

- a PFD step, which performs the actual diagnosis,
- an input design step, which aims to maximize the effectiveness of the PFD component.

While many existing AFD methods can be interpreted through this lens, the separation becomes particularly valuable in the RL setting. By treating the PFD block as a generic black box, FIERL can seamlessly integrate any PFD strategy into the optimization loop. This makes the framework broadly applicable and capable of tackling complex diagnostic scenarios, such as those involving a continuous spectrum of fault modes. Moreover, FIERL naturally extends to integrated AFDC, which is the main focus of this work, though the same ideas can be straightforwardly adapted to AFD without control.

7.2 System Modelling and Notation

A discrete-time stochastic system is commonly described by the set of equations

$$\begin{aligned} x_{t+1} &= g(x_t, u_t, w_t), \\ y_t &= h(x_t, u_t, v_t). \end{aligned} \tag{7.1}$$

where $x_t \in \mathbb{R}^{n_x}$, $y_t \in \mathbb{R}^{n_y}$ and $u_t \in \mathbb{R}^{n_u}$ represent respectively the system state, output and input at time t . The function $g(\cdot)$ describes the system dynamics while the function $h(\cdot)$ characterizes the measurement dynamics, i.e. the process that governs the model output. The stochastic variables $v_t \in \mathbb{R}^{n_v} \sim \mathcal{N}(\mu_v, \Sigma_v)$ and $w_t \in \mathbb{R}^{n_w} \sim \mathcal{N}(\mu_w, \Sigma_w)$ represent independent, typically stationary, Gaussian measurement and process noises and are used to capture stochasticity in the system evolution.

This description can be extended to also describe faulty systems, by considering time-varying faults $z_t \in Z$ in a generic fault set Z that can affect the dynamics and measurement of the system:

$$\begin{aligned} x_{t+1} &= g(z_t, x_t, u_t, w_t), \\ y_t &= h(z_t, x_t, u_t, v_t), \\ z_{t+1} &= l(z_t, x_t, u_t), \end{aligned} \tag{7.2}$$

In this extension the system dynamics and model output also depend on the fault value z_t . The fault free mode is given by a specific element $z^0 \in Z$ and the fault evolution dynamics are described by the stochastic function $l(\cdot)$.

In our example application, we consider linear systems with actuator faults, where Eq. 7.2 is reduced to:

$$\begin{aligned} x_{t+1} &= Ax_t + B \text{diag}(z_t)u_t + w_t \\ y_t &= Cx_t + v_t \\ z_{t+1} &= z_t + \xi_t \end{aligned} \tag{7.3}$$

where A , B and C are known real-valued matrices and each component of $z_t \in \mathbb{R}^{n_u}$ models partial faults of the corresponding actuator. Fault evolution is independent of x_t and u_t , and is characterized by the random variable ξ_t .

7.3 Constrained Reinforcement Learning

In basic Reinforcement Learning (RL) formulation, an *agent* interacts with its *environment*, trying to maximize some notion of *cumulative reward*. Basic sequential decision-making problems of this kind are usually modelled as Markov Decision Process (MDP). The structural components of an MDP are fully represented by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$, where the state space $\mathcal{S}$ and action space $\mathcal{A}$ are arbitrary sets. The environment state $S_t \in \mathcal{S}$ follows a discrete-time stochastic process, defined by the state transition function $\mathcal{T}(s, a, s') = \mathbb{P}[S_{t+1} = s' \mid S_t = s, A_t = a]$ which depends on the action

$A_t \in \mathcal{A}$ performed by the agent. Finally, the reward function $\mathcal{R}(s, a, s')$ is used to measure the quality of a given state transition. The behaviour of the agent is generally characterized by a *stochastic policy*, a function $\pi : \mathcal{S} \rightarrow \mathbf{P}(\mathcal{A})$ that specifies the probability that a given action will be selected given the environment state. The realization of an action state-trajectory $\tau = (s_0, a_0, s_1, a_1, \dots)$ obtained by following policy π is commonly denoted by $\tau \sim \pi$. The return (cumulative reward) represents the total reward (discounted by $\gamma \in (0, 1]$) accumulated by the agent over the trajectory τ . The goal of RL is to find the optimal policy amongst a set of admissible ones Π :

$$\pi^* = \arg \max_{\pi \in \Pi} \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t, s_{t+1}) \right]. \quad (7.4)$$

Many common algorithms also make use of the value function $V_\pi^\mathcal{R}$, the action-value function $Q_\pi^\mathcal{R}$ and the advantage function $A_\pi^\mathcal{R}$ defined as:

$$\begin{aligned} V_\pi^\mathcal{R}(s) &= \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t, s_{t+1}) \mid s_0 = s \right], \\ Q_\pi^\mathcal{R}(s, a) &= \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t, s_{t+1}) \mid s_0 = s, a_0 = a \right], \\ A_\pi^\mathcal{R}(s, a) &= Q_\pi(s, a) - V_\pi(s). \end{aligned} \quad (7.5)$$

Constrained Reinforcement Learning (CRL) addresses the general problem of training an RL agent under explicit constraints. The goal is to ensure that these constraints are satisfied not only during training but also at deployment, so that the agent consistently operates within predefined safety or performance boundaries. Beyond safety, CRL can also be applied to multi-objective problems, by treating one objective as the reward and reframing the others as constraints.

The standard mathematical formalism for CRL is the constrained Markov decision process (CMDP) [Alt99], which extends a classical MDP with a set of m cost functions,

$$C_i : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}, \quad i = 1, \dots, m. \quad (7.6)$$

Each cost function has the same structure as the reward function, allowing us to define its expected discounted return, known as the constraint return

(or C_i -return):

$$J_{C_i}(\pi) = \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{\infty} \gamma^t C_i(s_t, a_t, s_{t+1}) \right]. \quad (7.7)$$

The optimal policy in a CMDP is the solution to the constrained optimization problem:

$$\pi^* = \arg \max_{\substack{\pi: J_{C_i}(\pi) \leq d_i, \\ i=1, \dots, m}} J_{\mathcal{R}}(\pi), \quad (7.8)$$

where d_i are user-specified thresholds defining the admissible level for each constraint.

Equivalently, this can be viewed as solving a standard MDP but restricting the feasible policy set to

$$\Pi_C = \{\pi : J_{C_i}(\pi) \leq d_i, i = 1, \dots, m\}. \quad (7.9)$$

7.3.1 Constrained Policy Optimization

Constrained Policy Optimization (CPO) [AHTA17] is a general-purpose local policy search algorithm for CRL. It provides theoretical guarantees of near-constraint satisfaction throughout training and has proven effective for learning neural network policies in complex control tasks with high-dimensional constraints. A key advantage of CPO is that, unlike primal-dual optimization methods [CGJP17], it can enforce constraints during training without substantially degrading return performance. In fact, CPO generally outperforms both primal-dual approaches and fixed-penalty methods that incorporate constraints directly into the reward function [AHTA17].

As a local policy search method, CPO iteratively seeks an approximation to the optimal policy within a trust region. Given a parametrization θ of the policy space, the update is formulated as:

$$\begin{aligned} \pi_{\theta_{k+1}} &= \arg \max_{\pi \in \Pi} J(\pi), \\ \text{s.t. } D(\pi, \pi_{\theta_k}) &\leq \delta, \end{aligned} \quad (7.10)$$

where $J(\pi)$ is a surrogate objective and D is a distance measure on the policy space. The standard policy gradient update arises by linearizing J around π_{θ_k} and setting $D(\pi, \pi_{\theta_k}) = \|\theta - \theta_k\|_2$.

For Constrained Markov Decision Processes (CMDPs), the feasible domain Π_C consists of policies that satisfy the constraints. However, exact evaluation

of feasibility requires precise off-policy estimates, which is generally impractical. CPO circumvents this by introducing principled approximations of both the objective and the constraints based on the state distribution under the current policy ($s \sim \pi_{\theta_k}$). This yields the update:

$$\begin{aligned} \pi_{\theta_{k+1}} &= \arg \max_{\pi \in \Pi} \mathbb{E}_{a \sim \pi_{\theta_k}}^{s \sim \pi_{\theta_k}} [A^{\mathcal{R}} \pi_{\theta_k}(s, a)] , \\ \text{s.t.} \quad & J_{C_i}(\pi_{\theta_k}) + \frac{1}{1-\gamma} \mathbb{E}_{s \sim \pi_{\theta_k}}^{a \sim \pi_{\theta_k}} [A^{C_i} \pi_{\theta_k}(s, a)] \leq d_i, \\ & \mathbb{E} s \sim \pi_k [DKL(\pi_{\theta}(s) || \pi_{\theta_k}(s))] \leq \delta, \end{aligned} \quad (7.11)$$

where $A^{\mathcal{R}} \pi_{\theta_k}$ and $A^{C_i} \pi_{\theta_k}$ are the reward and cost advantages, respectively. The KL-divergence constraint makes CPO a trust region method, ensuring monotonic performance improvement and approximate constraint satisfaction [AHTA17]. This is especially critical with neural network policies, which can otherwise collapse after poor updates.

Directly solving Eq. 7.11 is intractable for high-dimensional parameter spaces. CPO therefore relies on linearizing both the objective and cost constraints around π_{θ_k} , while approximating the KL-divergence via a second-order expansion:

$$\begin{aligned} \theta_{k+1} &= \arg \max_{\theta} g^T(\theta - \theta_k), \\ \text{s.t.} \quad & J_{C_i}(\pi_{\theta_k}) + b_i^T(\theta - \theta_k) \leq 0, \quad i = 1, \dots, m, \\ & \frac{1}{2}(\theta - \theta_k)^T H(\theta - \theta_k) \leq \delta, \end{aligned} \quad (7.12)$$

where g is the gradient of the objective, b_i the gradient of constraint i , and H the Hessian of the KL-divergence (equivalently, the Fisher Information Matrix, which is positive semi-definite). This convex problem can be solved efficiently via duality.

Because approximations may yield infeasible updates, CPO includes a backtracking line search to ensure surrogate constraint satisfaction. If no feasible solution exists, a recovery step is performed. For the single-constraint case, a closed-form update can be applied:

$$\theta^* = \theta_k - \sqrt{\frac{2\delta}{b^T H^{-1} b}} H^{-1} b, \quad (7.13)$$

which guarantees a strict decrease in the constraint value.

Nevertheless, due to approximation errors, iterates may still become infeasible. A robust recovery strategy is therefore essential in practice. Full implementation details are discussed in [AHTA17].

7.4 Fault Identification Enhancement with Reinforcement Learning

In this work, the AFD paradigm is reinterpreted as an extension of Passive Fault Detection (PFD) approaches, by designing a feasible control strategy that optimizes passive detection accuracy. Indeed, AFD consists of two distinct tasks: (i) selecting a control action, and (ii) computing the fault detector output signal z_t^{est} . Deriving the optimal detector output from the available information is essentially a PFD problem. This explains why most AFD (and, by extension, AFDC) methods, when aiming to optimize a metric related to diagnostic accuracy, typically employ nested optimization loops. For example, many set-separation approaches rely on inner template-matching steps, which can be interpreted as implicit PFD.

In FIERL, we explicitly disentangle these two tasks: we delegate the PFD step to existing methods, while focusing on the input design, using RL to learn a control strategy that enhances the accuracy of the chosen PFD module. In particular, we propose the use of CRL to manage the conflicting goals of diagnostic accuracy and control quality. Rather than solving a dual-objective optimization problem, we directly optimize diagnosis accuracy subject to explicit constraints on control performance. This formulation is more intuitive in practice: setting bounds on control performance via hyperparameters d_i is generally easier than tuning weights to balance competing objectives.

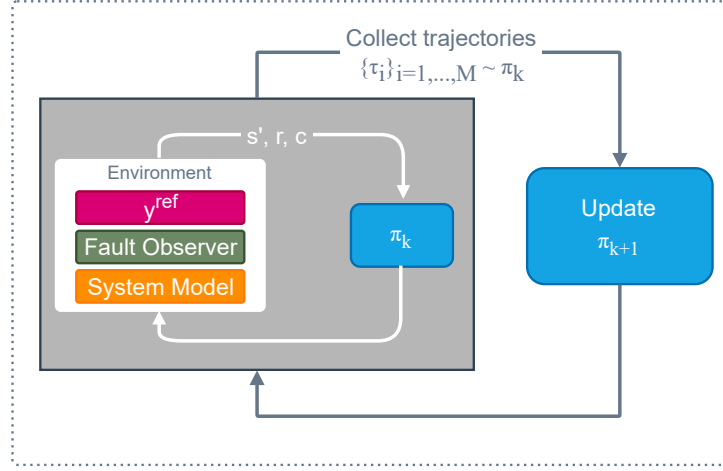


Figure 7.1: Depiction of the offline policy training phase for FIERL.

From these considerations, the AFD problem can be cast as a CMDP, where:

- Environment state. Includes the system state x , the true fault z , and passive estimates x_t^{est} and z_t^{est} . For AFDC, control-related signals such as the reference y_t^{ref} or output y_t must also be included.
- Agent action. Either the control input u_t (in integrated AFDC) or an auxiliary control signal (in pure AFD).
- Reward function. Defined by a metric $\mathcal{M}$ measuring how well the estimated fault signal z_t^{est} matches the true fault z_t :

$$\mathcal{R}(s_t, a_t, s_{t+1}) = \mathcal{M}(z_t^{est}, z_t). \quad (7.14)$$

- Cost function. Used to enforce guarantees on control performance. In this work, we focus on adherence of the system output y_t to a reference y_t^{ref} , measured by a distance $\mathcal{D}$:

$$\mathcal{C}(s_t, a_t, s_{t+1}) = \mathcal{D}(y_t^{ref}, y_t). \quad (7.15)$$

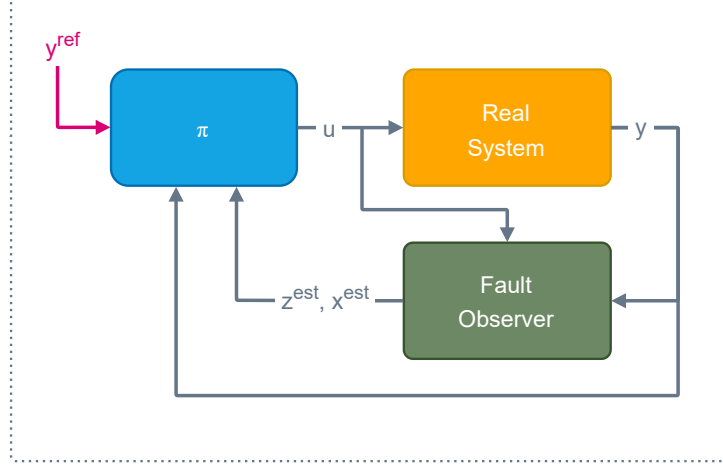


Figure 7.2: Depiction of the control flow during deployment of FIERL.

The resulting optimal policy takes the form:

$$\begin{aligned} \pi_{\theta^*} &= \arg \max_{\pi_{\theta}} \mathbb{E}_{\tau \sim \pi_{\theta}} \left[\sum_{t=0}^{\infty} \gamma^t \mathcal{M}(z_t^{est}, z_t) \right] . \\ \text{s.t. } \quad & \mathbb{E}_{\tau \sim \pi_{\theta}} \left[\sum_{t=0}^{\infty} \gamma_c^t \mathcal{D}(y_t, y_t^{ref}) \right] \leq d \end{aligned} \quad (7.16)$$

Since s_t must contain sufficient statistics for the environments future evolution, the system state x and true fault z are included in the formal definition. However, these are not observable to the agent. Thus, during both training and deployment, we operate with a masked partial state s'_t that excludes them. Figure ?? 7.2 illustrates this setup during training and deployment phases respectively.

7.5 Case Study

In this section, FIERL is evaluated on the three-tank benchmark system. The system model, expressed as in Eq. 7.3, is derived in the Appendix of [ZH20]. The state vector corresponds to the liquid levels in the three tanks, with system matrices obtained via linearization around the equilibrium point $x^* = [0.489, 0.2332, 0.3611]^T$ meters and discretized using a sampling period of $T_S = 0.1$ seconds.

Independent process and measurement noise are considered, with covariances $\Sigma_w = 10^{-8}$ and $\Sigma_v = 10^{-6}$, respectively. Specifically, the process

noise follows $w_t \sim \mathcal{N}(0, \text{diag}(10^{-6}, 10^{-6}, 10^{-6}))$, while the measurement noise is given by $v_t \sim \mathcal{N}(0, \text{diag}(10^{-6}, 10^{-6}))$.

The control input is $u_t \in \mathbb{R}^2$, constrained to $U = [-0.002, 0.02]$. The system output consists of the first two state components, reflecting the fact that the third component cannot be directly controlled.

7.5.1 Passive Component Derivation

For systems modelled by Eq. 7.3, assuming A , B and u_t to be known, a passive observer can be derived, which provides estimates for x_t and z_t at time τ of the form:

$$\begin{aligned} x_{t,\tau}^{est} &\sim \mathcal{N}(\mu_{x_t}^{(\tau)}, \Sigma_{x_t}^{(\tau)}) \\ z_{t,\tau}^{est} &\sim \mathcal{N}(\mu_{z_t}^{(\tau)}, \Sigma_{z_t}^{(\tau)}). \end{aligned} \quad (7.17)$$

Update rules for distribution parameters are derived in closed form using Bayesian inference. If omitted, assume $\tau = t$.

Consider $B_t^* = B \text{diag}(u_t)$ and Eq. 7.3 expressed as:

$$x_{t+1} = Ax_t + B_t^* z_t + w_t. \quad (7.18)$$

Direct substitution of the estimates at time t in Eq. 7.18 yields:

$$\begin{aligned} \mu_{x_{t+1}}^{(t)} &= A\mu_{x_t}^{(t)} + B_t^* \mu_{z_t}^{(t)} \\ \Sigma_{x_{t+1}}^{(t)} &= A\Sigma_{x_t}^{(t)} A^T + B_t^* \Sigma_{z_t}^{(t)} B_t^{*T} + \Sigma_w. \end{aligned} \quad (7.19)$$

To account for y_{t+1} , the density term $p(x_{t+1} | y_{t+1})$ can be directly evaluated applying Bayes' theorem, resulting in:

$$\begin{aligned} \Sigma_{x_{t+1}}^{(t+1)} &= ((\Sigma_{x_{t+1}}^{(t)})^{-1} + C^T \Sigma_v^{-1} C)^{-1} \\ \mu_{x_{t+1}}^{(t+1)} &= \Sigma_{x_{t+1}}^{t+1} ((\Sigma_{x_{t+1}}^{(t)})^{-1} \mu_{x_{t+1}}^{(t)} + C^T \Sigma_v^{-1} y_{t+1}). \end{aligned} \quad (7.20)$$

Applying Woodbury matrix identity [Woo50] and performing some algebraic manipulations, a more numerically stable set of equations can be obtained:

$$\begin{aligned} K_x(t) &= \Sigma_{x_{t+1}}^{(t)} C^T (\Sigma_v + C \Sigma_{x_{t+1}}^{(t)} C^T)^{-1} \\ \Sigma_{x_{t+1}} &= (I - K_x(t) C) \Sigma_{x_{t+1}}^{(t)} \\ \mu_{x_{t+1}} &= \mu_{x_{t+1}}^{(t)} + K_x(t) (y_{t+1} - C \mu_{x_{t+1}}^{(t)}). \end{aligned} \quad (7.21)$$

The procedure up to this point is equivalent to the derivation of a standard Kalman filter, where the uncertainty in the fault vector z_t adds to the existing process noise.

Parameter update rules for the fault estimate can also be derived from Eq. 7.18. Substituting best estimates for all terms that do not depend on z_t yields:

$$B_t^* z_t \sim N(\mu_{x_{t+1}}^{(t+1)} - A\mu_{x_t}^{(t)}, \Sigma_{x_{t+1}}^{(t+1)} + A\Sigma_{x_t}^{(t)}A^T + \Sigma_w) \quad (7.22)$$

This represents a way to measure z_t , analogously to how the term $Cx_t \sim N(y_t, \Sigma_w)$ represents a measurement for x_t . Therefore, using the previous results in Eq. 7.21 and substituting the correct mean and variances:

$$\begin{aligned} K_z(t) &= \Sigma_{z_t}^{(t)} B_t^{*T} (\Sigma_{x_{t+1}}^{(t+1)} + \Sigma_{x_{t+1}}^{(t)})^{-1} \\ \Sigma_{z_t}^{(t+1)} &= (I - K_z(t)B_t^*)\Sigma_{z_t}^{(t)} \\ \mu_{z_t}^{(t+1)} &= \mu_{z_t}^{(t)} + K_z(t)(\mu_{x_{t+1}}^{(t+1)} - \mu_{x_{t+1}}^{(t)}). \end{aligned} \quad (7.23)$$

Note that this improves the estimate for the parameter at time t , using information obtained from y_{t+1} at the following time step. To obtain $z_{t+1}^{(t+1)}$, some assumptions on the fault evolution need to be made. In the case of $\xi_t \sim \mathcal{N}(\mu_{\xi_t}, \Sigma_{\xi_t})$ the parameter updates become:

$$\Sigma_{z_{t+1}}^{(t+1)} = \Sigma_{z_t}^{(t+1)} + \Sigma_{\xi_t}, \quad \mu_{z_{t+1}}^{(t+1)} = \mu_{z_t}^{(t+1)} + \mu_{\xi_t}. \quad (7.24)$$

Notably, when $\mu_{\xi_t} = 0$, the update reduces to adding a bias term to the estimate covariance. This simple addition produces remarkably stable filters that can perform reasonably even if the fault dynamics do not match the assumptions (i.e. jump processes) as shown in 7.5.3.

7.5.2 Framework Application

Since the considered passive component provides estimates in the form of Gaussian distributions, the selected reward metric is the negative expected squared norm of the fault estimation error, which admits the closed form:

$$\begin{aligned} \mathcal{R}(s_t, a_t, s_{t+1}) &= -\mathbb{E}_{z \sim z_t^{est}} (\|z_t^{true} - z\|_2^2) \\ &= -\text{trace}(\Sigma_{z_t}) - \|z_t^{true} - \mu_{z_t}\|_2^2. \end{aligned} \quad (7.25)$$

The cost function counts the number of times the adherence to the reference exceeds a desired tolerance Δy^{max} :

$$C(s_t, a_t, s_{t+1}) = \begin{cases} 0 & \text{if } \|y_t - y_t^{ref}\|_\infty \leq \Delta y^{max} \\ 1 & \text{if } \|y_t - y_t^{ref}\|_\infty > \Delta y^{max} \end{cases}. \quad (7.26)$$

The full state vector incorporates the state and fault estimate distribution parameters, the system output and reference:

$$s'_t = [\mu_{x_t} \quad \text{triu}(\Sigma_{x_t}) \quad \mu_{z_t} \quad \text{triu}(\Sigma_{z_t}) \quad y_t^{ref} \quad y_t]. \quad (7.27)$$

Therefore the optimal policy becomes:

$$\begin{aligned} \pi_{\theta^*} = \arg \max_{\pi_{\theta}} \mathbb{E}_{\tau \sim \pi_{\theta}} \left[\sum_{t=0}^{\infty} -\gamma^t \mathbb{E}_{z \sim z_t^{est}} (\|z_t^{true} - z\|_2^2) \right] \\ \text{s.t.} \quad \mathbb{E}_{\tau \sim \pi_{\theta}} \left[\sum_{t=0}^{\infty} \gamma_c^t \mathbf{1}_{(\Delta y^{max}, \infty)} (\|y_t - y_t^{ref}\|_{\infty}) \right] \leq d \end{aligned} \quad (7.28)$$

In this case the parameter d represents the allowed tracking bound violation rate. For $d = 0$ it must be enforced at all times.

7.5.3 Experimental results

To evaluate its effectiveness, the FIERL policy is compared against a naive baseline consisting of a proportional controller with randomly perturbed control inputs:

$$\Delta u_t^P \sim \mathcal{U}([-K_p, K_p]). \quad (7.29)$$

The proportional gain K_p is tuned to achieve the best possible diagnosis performance while respecting the same tracking constraints imposed on the RL policy.

Despite its simplicity, this approach is surprisingly effective. The proportional controller ensures acceptable control performance, while the injected perturbations provide additional information about the fault state. In fact, if the input domain were symmetric (as originally proposed in [ZH20]), this naive strategy would approximate the optimal policy remarkably well, making the benefits of FIERL harder to discern. This observation motivated the modification introduced in this study.

The proportional gain was tuned to $K_p = 0.016$, yielding the highest average return without exceeding the tracking violation budget $d = 6$, based on simulations with single, uniformly sampled faults persisting across 40 steps.

The RL agent is trained for 1000 policy updates, each using data from 90 short episodes of length $N_D = 40$ steps (4 seconds). At the start of each episode, the true fault vector is sampled uniformly from $[0, 1] \times [0, 1]$ and kept

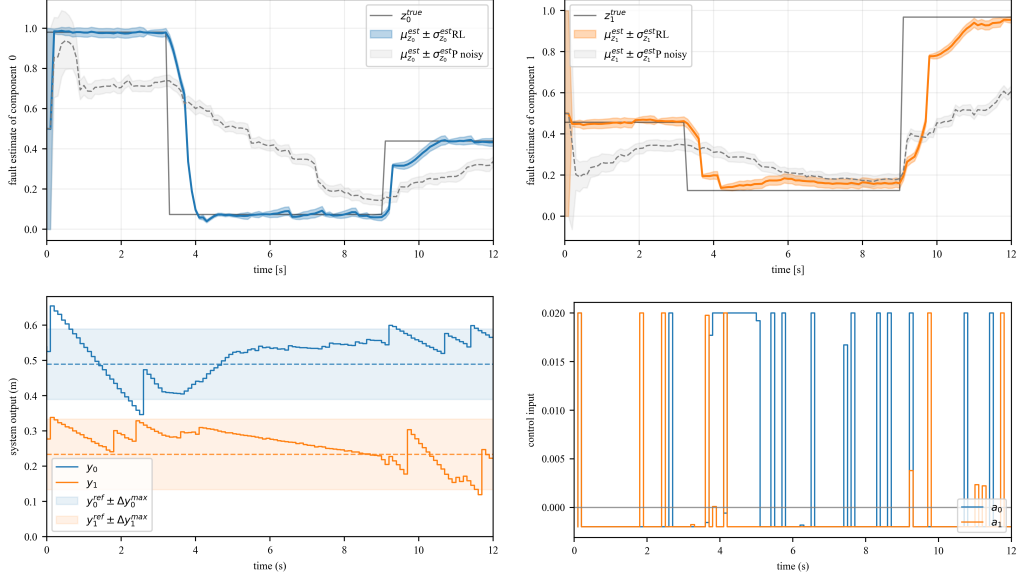


Figure 7.3: Experimental results for a randomly selected testing episode. The top two panels compare the evolution of fault estimate distributions for FIERL against the naive approach.

constant throughout the episode. The initial liquid level is sampled uniformly within a ball of radius $\Delta y^{max} = 0.1$ centered at x^* . The cost constraint is set to $d = 6 = 0.15N_D$. The passive component is initialized with an uninformative prior $z_0^{est} \sim \mathcal{N}((0.5, 0.5), \text{diag}(1, 1))$, and the estimated covariance bias is set to $\Sigma_{\omega_t} = 10^{-3}$.

For testing, long episodes are used with durations between 90 and 180 steps (instead of the 40 used for training). Fault trajectories change abruptly at random time steps, with each fault persisting for at least 30 steps to ensure sufficient time for reliable estimation.

Averaging over 10,000 test episodes, the RL policy achieves an expected return per step of

$$R_{RL} = (-1.458 \pm 0.562) \cdot 10^{-2}, \quad C_{RL} = (1.178 \pm 1.145) \cdot 10^{-1}, \quad (7.30)$$

while the proportional controller yields significantly worse performance:

$$R_P = (-3.892 \pm 1.477) \cdot 10^{-2}, \quad C_P = (0.030 \pm 2.18) \cdot 10^{-3}. \quad (7.31)$$

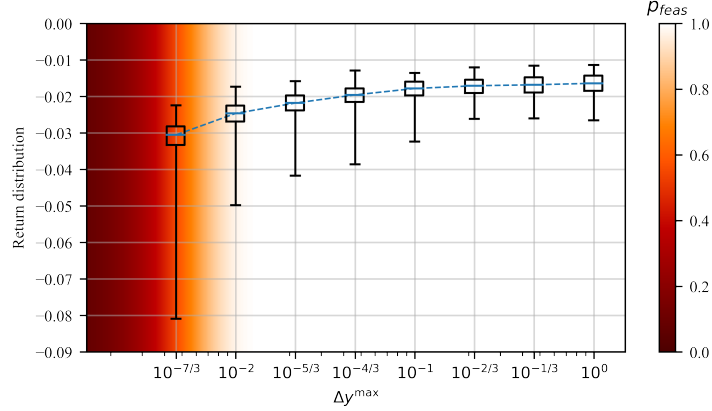


Figure 7.4: Diagnosis performance as a function of the tracking threshold $\Delta y^{\max}$. Each boxplot represents the distribution of average returns across test episodes for FIERL under the corresponding control requirements. The colormap illustrates the probability of constraint violations caused solely by process noise drifts.

Figure 7.3 illustrates the behavior in a representative test episode. With the RL-based policy, the fault estimates converge rapidly to the true fault values after jumps, remain generally more accurate, and exhibit tighter distributions overall. Due to the passive component design, short bursty actions yield more information about the fault mode than applying the same control effort gradually over time. This behavior is consistent with the RL policy shown in the fourth panel, where a sharp control action is followed by a sequence of small compensating negative inputs, ensuring that the tracking requirements (depicted in panel 3) are still satisfied.

On the other hand Figure 7.4 highlights the trade-off between tracking constraints and diagnosis accuracy. As $\Delta y^{\max}$ decreases, diagnosis performance drops sharply and training becomes unstable. This occurs because the initial policy, regardless of how conservatively it is initialized, is almost guaranteed to be infeasible, a condition necessary to satisfy the convergence guarantees of [AHTA17]. The code for reproducing these experiments is publicly available at: github.com/davidesartor/FIERL.

7.6 Conclusions

This work introduced a general application of CRL to enhance the performance of passive diagnosis modules. The integration of FIERL with the application-specific PFD component derived in Section 7.5.1 highlights several important features of the method. Crucially, it does not require perfect knowledge of the system state, relying instead on state estimates. Unlike much of the existing literature, this strategy can effectively handle a continuous spectrum of faults, or more generally, any stochastic fault dynamics that can be realistically simulated. Moreover, FIERL can in principle be applied directly to non-linear, time-varying systems. While this may result in more complex PFD modules, it does not affect the structure of the RL loop itself. Another important feature is robustness to unseen fault dynamics: as shown in Section 7.5.3, even when trained only on stationary faults, FIERL demonstrated effectiveness in handling transient ones. This is particularly relevant in continuous monitoring scenarios.

On the other hand, RL methods remain computationally intensive, requiring accurate system models and simulations to capture the correct dynamics. Determining convergence is also non-trivial: it is difficult to assess whether learning has converged across the full state space, and in practice full convergence cannot be guaranteed. Theoretical guarantees exist only under the assumption of perfect knowledge of the value function, which is unrealistic in real-world applications [NJG17].

A further limitation lies in the cost formulation. In Section 7.5.3, the cost was defined to limit the expected number of constraint violations to at most d occurrences on average across episodes. While effective in expectation, this formulation may lead to objective misalignment: the optimal policy could substantially exceed the violation limit in some regions of the state space, compensating with negligible violations in others. This issue is not due to failed convergence but to a mismatch between the intended and encoded objectives, potentially resulting in undesirable behaviors.

Future research directions include refining the cost design, for example by replacing the indicator function with penalties that discourage repeated violations within a single episode, and extending evaluation to more challenging scenarios such as non-linear or over-actuated systems, as well as more general fault dynamics.

8

Hierarchical Abstractions for Generation of Options

8.1 Introduction

Reinforcement Learning (RL) has achieved remarkable success across a wide range of decision-making tasks. However, its effectiveness diminishes in environments with large state spaces and sparse rewards. As the number of possible states grows and the temporal horizon extends, standard RL methods often struggle to discover effective strategies. This challenge has motivated the development of Hierarchical Reinforcement Learning (HRL), which decomposes complex problems into smaller, more manageable subtasks [Die99, SPS99, H⁺02].

Hierarchy in artificial intelligence has long been a central theme [Sac74]. Within RL, hierarchy is commonly realized by decomposing an overarching Markov Decision Process (MDP) into multiple sub-MDPs [SPS99, SC12, KB09]. These sub-MDPs capture meaningful subtasks defined by core states or transitions. A useful distinction is between temporal abstraction, which emphasizes critical subtrajectories, and state-action abstraction, which compresses the original state space and represents subtrajectories as transitions

in an abstract domain [HAGL19, HAC⁺21]. Although these two forms are often presented separately, they are not mutually exclusive and frequently interact in practice [Die99, RB03, AUK⁺20].

Much of the early work in HRL focused on temporal-difference methods, which break tasks into temporal lags defining specific subgoals. These subgoals can then be combined into macro-subgoals, enabling agents to identify and achieve intermediate objectives on the way to the main task [SPS99, SC12, KB09]. More recently, formal frameworks for abstraction have been developed. For example, [AUK⁺20] provide a mathematical treatment of state and action abstractions, introducing methods to combine state abstractions with abstract actions (also known as options [SPS99]) via bisimulation metrics. Comprehensive reviews, such as [Abe22], highlight the diversity of approaches and their respective trade-offs, while related works [RN18, KKL15] demonstrate how symbolic and state abstractions can be integrated effectively.

To address the limitations of standard RL in sparse-reward, long-horizon settings, we propose MANGO (Multilayer Abstraction for Nested Generation of Options), a novel hierarchical RL framework. MANGO decomposes tasks into multiple abstraction layers, where each layer defines its own abstract state space and organizes behavior into modular macro-actions, or options. These options are nested across layers, enabling efficient reuse of learned skills and improved sample efficiency. Within each layer, intra-layer policies guide transitions in the abstract space, while task actions incorporate task-specific components such as reward functions.

Building on the principles of state-action abstraction [Die99, AUK⁺20], MANGO introduces functions that synthesize and abstract features of the environment hierarchically. Each layer corresponds to a specific feature-state space, within which options are defined as abstract state transformations. To accelerate learning, we first perform a task-independent discovery of options, as suggested in [VZH⁺21], and then define an RL environment where the agent executes options rather than primitive actions.

Experiments in procedurally generated grid environments demonstrate that MANGO significantly improves both sample efficiency and generalization compared to standard RL baselines. Beyond raw performance, the framework enhances interpretability: by decomposing decisions across multiple layers, MANGO provides transparency about which subtask or higher-level goal drives the agents behavior at each timestep. This level of interpretability is especially valuable in industrial and safety-critical domains,

where understanding and, if necessary, correcting an agents rationale is as important as overall task performance.

8.2 Options in the Literature

Options, introduced by [SPS99], provide a structured mechanism for incorporating abstract behaviors into an agents decision process. Conceptually, options are learned subroutines that the agent can invoke just like primitive actions, thereby extending and enriching the agents action space.

Traditionally, an option is defined by three components: (i) a policy that specifies the behavior of the agent while the option is active, (ii) an initialization condition that determines the set of states where the option can be initiated, and (iii) a termination condition that dictates when the option concludes.

In our framework, we adopt a simplified formulation by omitting the initialization condition. We assume that any option can be initiated from any state of the environment, focusing instead on the policy and termination condition to define its behavior.

Definition 8.2.1 (Options). Options consist of two components $o: = \langle \pi_o, \beta_o \rangle$:

- a policy $\pi_o: \mathcal{S} \rightarrow \mathbb{P}(\mathcal{A})$ guiding actions within states,
- a condition $\beta_o: \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \{False, True\}$ defining if the option should terminate after a transition.

The *option space* is defined as the set of available options:

$$\mathcal{O} = \{o_1, o_2, \dots, o_k\},$$

where each o_i is an option as previously defined. This expands the action space of the agent from primitive actions $\mathcal{A}$ to a combined action-option space $\mathcal{A}' = \mathcal{A} \cup \mathcal{O}$.

Consequently, the agent's policy π becomes hierarchical, mapping states to distributions over this expanded space:

$$\mu: \mathcal{S} \rightarrow \mathbb{P}(\mathcal{A}'), \tag{8.1}$$

thus enabling the agent to choose either a primitive action or initiate an option at each step. When an option $o \in \mathcal{O}$ is selected, the agent follows the options internal policy π_o until the termination condition β_o evaluates to *True*, after which the agent resumes selecting from the expanded action-option space.

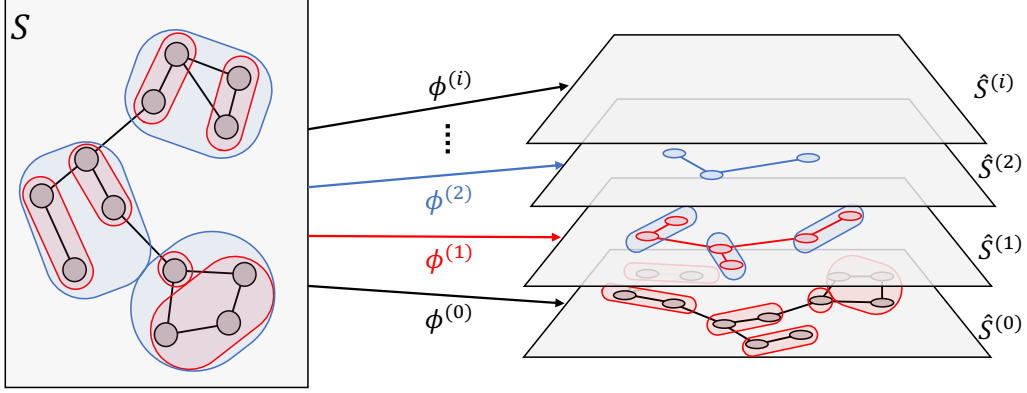


Figure 8.1: Progressive compression of the environment graph given a hierarchical abstraction.

8.3 Options as Actions in Abstract Graph

The state space S and the transition function $\mathcal{T}$ define a directed graph, where nodes are represented by the states of the MDP and the weight of the edges is given by the transition probability. In order to obtain an abstract, simplified representation of this graph, we introduce the notion of Concept Function.

Definition 8.3.1 (Concept Function). A concept function is a function denoted by $\phi : S \rightarrow S^\phi$, where S^ϕ is a connected partition of the state space S , that is, the induced sub-graphs are connected graphs.

A Concept Function should capture salient environmental attributes in a coarse, abstract representation S^ϕ . We insist that the partitions induced by concept functions must be connected. This prerequisite is vital for eliminating ambiguity in an agent’s movements within the environment. In this study, we presume that prior knowledge allows the derivation of these concepts, enabling their direct manual definition.

In an MDP, an action a is represented by the transition dynamics through the transition function $\mathcal{T}_a$. With the same objective in mind, we define an abstract action as a^ϕ to be represented by a transition function $\mathcal{T}_{a^\phi} : S^\phi \rightarrow \mathbb{P}(S^\phi)$. The abstract action a^ϕ represents the target transition dynamic that the agent aims to achieve in the abstract state space S^ϕ .

Similarly to actions, options can be associated with an empirical transition function $\mathcal{T}_o(s, s')$, which describes the probability that option o ends in state s' given that it was initiated in state s . The multi-step trajectory performed by an option o can be interpreted as one-step in the abstract state space $\mathcal{S}^\phi$.

Consequently, we establish a one-to-one correspondence between abstract actions and options, denoting each option characterized by a specific abstract action a^ϕ as o_{a^ϕ} . For this purpose, we will formalize the aim of options as minimizing the dissimilarity between the target transition $\mathcal{T}_{a^\phi}$ and the empirical transition $\mathcal{T}_{o_{a^\phi}}$.

A single concept function can be associated with multiple actions, discerning the abstract action space

$$\mathcal{A}^\phi = \{a_1^\phi, a_2^\phi, \dots, a_k^\phi\}.$$

This, in turn, defines the corresponding option space

$$\mathcal{O}^\phi = \{o_{a_1^\phi}, o_{a_2^\phi}, \dots, o_{a_k^\phi}\}.$$

A Concept Function ϕ can also be used to naturally define a termination condition β^ϕ , which triggers when the concept changes, i.e., when the agent transitions from one abstract state to another. This termination condition is shared across all options in $\mathcal{O}^\phi$ and can be formally defined as:

Definition 8.3.2 (Concept Termination condition). Given a concept function ϕ , the termination condition β^ϕ is defined as:

$$\beta^\phi(s, a, s') = \begin{cases} True & \text{if } \phi(s) \neq \phi(s') \\ False & \text{otherwise} \end{cases} \quad (8.2)$$

8.4 Structured Options using Hierarchical Abstractions

Traditionally, options are defined directly in terms of atomic environment actions. However, allowing options to be defined in terms of other options can

To simplify the discussion, we only consider the case where the number of options is the same for all concepts, but in principle, each concept could be associated with a different number of options.

lead to a more structured and efficient learning process. In our framework, we organize options in a multi-layer hierarchy, in order to obtain a structured and well-behaved option dependency graph.

Definition 8.4.1 (Hierarchical Abstraction). A set of concept functions $\phi_0, \phi_1, \dots, \phi_n$ defines a hierarchical abstraction if the respective state space partitions $\mathcal{S}^{\phi_i}$ are progressive refinements of one another. That is, $\mathcal{S}^{\phi_i} \supset \mathcal{S}^{\phi_{i+1}}$ for all $i \in (0, 1, \dots, n-1)$.

A concept function that is part of a hierarchical abstraction will be denoted as an Abstraction Function.

The refinement relation is a partial order in the set of all partitions, therefore not all combination of concept functions can be organized as hierarchical abstractions.

The identity function is the most trivial example of a concept function, where the state space is partitioned into singletons. This is useful for defining the base layer of the hierarchy, where the agent can perform atomic actions in the environment. Therefore, we define the base layer of the hierarchy ϕ_0 to be the identity, meaning that $\mathcal{S}^{\phi_0} = \mathcal{S}$.

It is also convenient to slightly modify the shared termination condition β^{ϕ_0} to be always true, instead of triggering only when the state changes. This way, options at this layer of abstraction are directly connected to atomic actions in the environment.

By requiring that partitions induced by abstraction functions are refinements of one another, we ensure that edges in higher-order graphs can be decomposed into multiple movements between lower-level states.

This allows options at a higher abstraction level i can be defined in terms of options at lower levels $j < i$. That is, the policy of a high-level option can invoke lower-level options as its actions, resulting in a nested, hierarchical structure. This recursive composition enables complex behaviors at higher layers to be constructed from simpler, reusable subroutines at lower layers, facilitating efficient learning and transfer across tasks.

Definition 8.4.2 (Intra-layer Policy). An *intra-layer policy* for the option $o \in \mathcal{O}^{\phi_i}$ is formally defined as

$$\hat{\pi}_{o^{\phi_i}} : \mathcal{S} \rightarrow \mathbb{P}(\mathcal{O}^{\phi_{i-1}})$$

and defines how options at the lower layer of abstraction can be combined to obtain the desired behaviour.

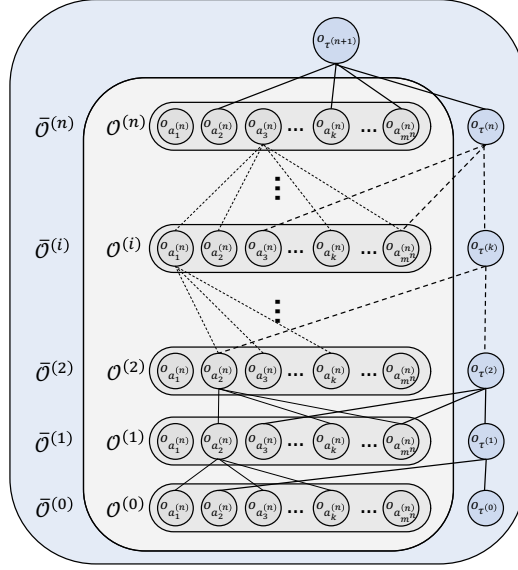


Figure 8.2: Schematic representation of the abstract action sets and their interconnection, including the task options at each level of abstraction τ^{ϕ_i} .

The intra-layer policy $\hat{\pi}_{o^{\phi_i}}$ implicitly defines routine associated to the option o^{ϕ_i} , just like the agent policy in 8.1 is used to express complex agent behavior in terms of simpler options. The option routine is obtained by selecting an option in $\mathcal{O}^{\phi_{i-1}}$. This lower-level option is followed until the termination condition $\beta_t^{\phi_{i-1}}$ is achieved, i.e., a state transition is executed at the lower abstraction layer $\mathcal{S}^{\phi_{i-1}}$. After the transition is completed, once again the policy $\hat{\pi}_{o^{\phi_i}}$ provides a probability distribution that leads to the selection of the next option.

8.5 Task actions

Solving a particular task in hierarchical RL amounts to maximizing the expected return in Equation ?? accumulated following the agent policy μ .

Unfortunately, given n layers of abstractions $\phi_0, \phi_1, \dots, \phi_n$, each with option set of size $|\mathcal{O}^{\phi_i}| = k$, yields a combined options set of size $(n + 1) \cdot k$. Thus, the size of the option set obtained using the proposed framework grows relatively rapidly as more and more abstraction layers are considered, making the direct optimization of the policy μ increasingly difficult. However, the

agent policy μ can be restructured in a way that takes advantage of the hierarchical organization of the option space: instead of a single policy over a large option space, a hierarchy of simpler policies is used.

Similarly to the hierarchical organization of options, the agent's policy can be arranged within the hierarchy as an additional abstract action integrated to each layer with an additional instantiation of an upper layer ($n + 1$) that will move the agent using the highest level options.

Definition 8.5.1 (Task Actions and Policies). A *task action* τ^{ϕ_i} is an abstract action defined at any layer $i \in \{0, \dots, n, n + 1\}$. The corresponding expanded abstract action space is given by $\bar{\mathcal{A}}^{\phi_i} := \mathcal{A}^{\phi_i} \cup \{\tau^{\phi_i}\}$.

Each task action τ^{ϕ_i} induces a corresponding *task policy*, denoted by $\pi_{\tau^{\phi_i}} : \mathcal{S} \rightarrow \mathbb{P}(\bar{\mathcal{A}}^{\phi_{i-1}})$, which operates over the expanded action space of the layer below. These policies, analogous to those associated with other abstract actions, capture the hierarchical structure of decision-making. Due to the temporal abstraction introduced by this recursive structure, task policies are defined as *semi intra-layer policies* $\hat{\pi}_{\tau^{\phi_i}}$.

While task actions mirror abstract actions structurally, they differ in termination: abstract actions end on an abstract state change, whereas task options terminate without inducing any state change at layer ϕ_i . Execution cascades upward, allowing higher-layer intra-layer policies to resume execution, and terminates upward, layer by layer, until reaching the top-level option.

The resultant subtrajectories are then linked either to previously trained options characterized by abstract actions $a^{\phi_{i-1}}$ or to task options $o_{\tau^{\phi_i}}$ if identified by $\tau^{\phi_{i-1}}$.

The termination function is given by the following.

Definition 8.5.2 (Task option termination function). Given the task option τ^{ϕ_i} , the termination function $\beta^{\tau^{\phi_i}}$ is defined as:

$$\beta_{\tau^{\phi_i}}(s_t, \bar{a}_t^{\phi_{i-1}}, s_t) = \begin{cases} \text{True if } \bar{a}_t^{\phi_{i-1}} = \tau^{\phi_{i-1}} \\ 0 \text{ otherwise} \end{cases}$$

except for $o_{\tau^{(0)}}$, which is the null option (empty routine)

8.6 Learning Process

We defined a generic option $o_{\bar{a}^{\phi_i}}$ associated with the action $\bar{a}^{\phi_i} \in \bar{\mathcal{A}}^{\phi_i}$ as a routine in the base layer, where $\bar{a}^{\phi_i}$ is an abstract action or a task action. This

option routine comprises a chain of semi-intra-layer policies, which navigate through the states of the (j) -th level of abstraction with $j < i$ until the termination conditions are encountered.

An MDP in the layer of abstraction ϕ_i consists of the set of states $\mathcal{S}$ introduced in Section 9.3, the abstract actions $\mathcal{A}^{\phi_i}$ introduced in Section 8.2, the group of transition function $\mathcal{T}_{a^{\phi_i}}$ with $a^{\phi_i} \in \mathcal{A}^{\phi_i}$ introduced in Section 8.2 and a specific discount factor γ^{ϕ_i} . Given that the MDP provide the structure to train the specific policy that replicates the trajectory within layer ϕ_i when executing an action $a^{\phi_{i+1}}$ from the layer above, we characterize the reward associated to this specific action defining it by $R_{a^{\phi_{i+1}}}$. Since there are multiple actions from the layer above $a^{\phi_{i+1}} \in \mathcal{A}^{\phi_{i+1}}$, corresponding to different options $o_{a^{\phi_{i+1}}}$ with their intra-layer policies in layer ϕ_i , we create distinct MDPs within layer ϕ_i with identical structural elements $(\mathcal{S}, \mathcal{A}^{\phi_i}, \{\mathcal{T}_{a^{\phi_i}} | a^{\phi_i} \in \mathcal{A}^{\phi_i}\}, \gamma^{\phi_i})$ but varying reward functions aligned with the abstract actions from the layer above.

Definition 8.6.1 (SMDP of the action a^{ϕ_i}). Given the two Layer of a Hierarchical Abstraction ϕ_i and ϕ_{i+1} , given an action $\bar{a}^{\phi_{i+1}} \in \bar{\mathcal{A}}^{\phi_{i+1}}$ we define the Markov Decision Process of that action as the tuple of: structural objects, the discount factor and the reward functions that shape the environment of the layer ϕ_i according to given action $\bar{a}^{\phi_{i+1}}$, $(\mathcal{S}, \bar{\mathcal{A}}^{\phi_i}, \{\mathcal{T}_{\bar{a}^{\phi_i}} | \bar{a}^{\phi_i} \in \bar{\mathcal{A}}^{\phi_i}\}, \gamma^{\phi_i}, R_{\bar{a}^{\phi_{i+1}}})$. $R_{\bar{a}^{\phi_{i+1}}} : \mathcal{S} \times \bar{\mathcal{A}}^{\phi_i} \times \mathcal{S} \rightarrow \mathbb{R}$ is the reward function such that $R_{\bar{a}^{\phi_{i+1}}}(s_t, a_t^{\phi_i}, s_{t+1})$ is given by:

$$\sum_{(s_k, a_k, s_{k+1}) \sim o_{\bar{a}_t^{\phi_i}}} \gamma_{\phi_i}^k r_{\bar{a}^{\phi_{i+1}}}^k(s_k, a_k, s_{k+1}) \quad (8.3)$$

and $r_{\bar{a}^{\phi_{i+1}}}(s_k, a_k, s_{k+1})$ is defined as:

$$\begin{cases} R(s_k, a_k, s_{k+1}) & \text{if } \bar{a}^{\phi_{i+1}} = \tau^{\phi_{i+1}} \\ \beta_k^{\phi_{i+1}} (1 - 2d_{\mathcal{T}_{a^{\phi_{i+1}}}}(\delta(\phi^{\phi_i}(s_k), \phi^{\phi_i}(s_{k-1})))) & \text{if } \bar{a}^{\phi_{i+1}} = a^{\phi_{i+1}} \end{cases} \quad (8.4)$$

The function $r_{\bar{a}^{\phi_{i+1}}}(s_k, a_k, s_{k+1})$ takes into consideration the current state s_k , the action a_k taken, and the subsequent state s_{k+1} , all contingent on the macro action $\bar{a}^{\phi_{i+1}}$ at hierarchy level $i + 1$. When $\bar{a}^{\phi_{i+1}}$ corresponds to a task action $\tau^{\phi_{i+1}}$, the reward is derived from the standard reward function $R(s_k, a_k, s_{k+1})$. However, if $\bar{a}^{\phi_{i+1}}$ represents an abstract action $a^{\phi_{i+1}}$,

the reward is intrinsic in the structure and influenced by the abstract transition generated by the trajectory. It is nonzero only if the trajectory terminates, which is indicated by $\beta_{k+1}^{\phi_{i+1}} = 1$. Additionally, the reward is adjusted based on the alignment of the occurred abstract transition with the intended transition defined by the abstract action dynamics, quantified by $(1 - 2d\tau_{a_{\phi_{i+1}}}(\delta(\phi^{\phi_i}(s_k), \phi^{\phi_i}(s_k - 1))))$.

It is then possible to explicitly write the Q-function related to the training of the policy of the an option at a given layer ϕ_i as $Q_{\bar{a}^{\phi_i}}(s_t, o_{\bar{a}_t^{\phi_{i-1}}})$ where $\bar{a}_t^{\phi_{i-1}} \in \bar{\mathcal{A}}^{\phi_{i-1}}$:

$$Q_{\bar{a}^{\phi_i}}(s_t, \bar{a}_t^{\phi_{i-1}}) = \mathbb{E}_{s_{t+1} \sim \pi_{o_{\bar{a}_t^{\phi_{i-1}}}}} [r_{\bar{a}^{\phi_i}}(s_t, \bar{a}_t^{\phi_{i-1}}, s_{t+1}) + \gamma^{\phi_{i-1}} \max_{\bar{a}_{t+1}^{\phi_{i-1}} \in \bar{\mathcal{A}}^{\phi_i}} Q(s_{t+1}, \bar{a}_{t+1}^{\phi_{i-1}})] \quad (8.5)$$

Here, $Q_{\bar{a}^{\phi_i}}(s_t, \bar{a}_t^{\phi_{i-1}})$ represents the expected cumulative reward gained by the policy $\bar{a}^{\phi_i}$ starting from state s_t and executing the option characterized by the action $\bar{a}_t^{\phi_{i-1}}$. The expectation is taken over the state distribution s_{t+1} at the end of the policy characterized by the action $\bar{a}_t^{\phi_{i-1}}$. The expression $r_{\bar{a}^{\phi_i}}(s_t, o_{\bar{a}_t^{\phi_{i-1}}}, s_{t+1})$ denotes the reward shaped by $\bar{a}^{\phi_i}$ obtained by executing option $o_{\bar{a}_t^{\phi_{i-1}}}$ at state s_t , and $\gamma^{\phi_i} \cdot \max_{\bar{a}_{t+1}^{\phi_{i-1}} \in \bar{\mathcal{A}}^{\phi_i}} Q(s_{t+1}, \bar{a}_{t+1}^{\phi_{i-1}})$ is the discounted maximum expected future reward from the next state s_{t+1} , considering the abstract actions available for the subsequent step $\bar{\mathcal{A}}^{\phi_i}$.

After training various task actions at each layer, the top-level task option $\tau^{\phi_{i+1}}$ can then act as the agent's policy, represented as μ . This policy oversees the entire process, utilizing the options to orchestrate the agent's movements and terminates with the conclusion of the environment's entire process.

8.7 Experimental Results

We consider a grid-based environment inspired by Frozen Lake. Each cell is either frozen (safe) or a hole (which ends the episode immediately). The agent begins in a randomly chosen cell, while the goal (the gift) is placed at another random location. The agent receives a reward upon reaching the goal, and each episode terminates either on success (reaching the goal), failure (falling into a hole), or when a maximum step limit is exceeded. The

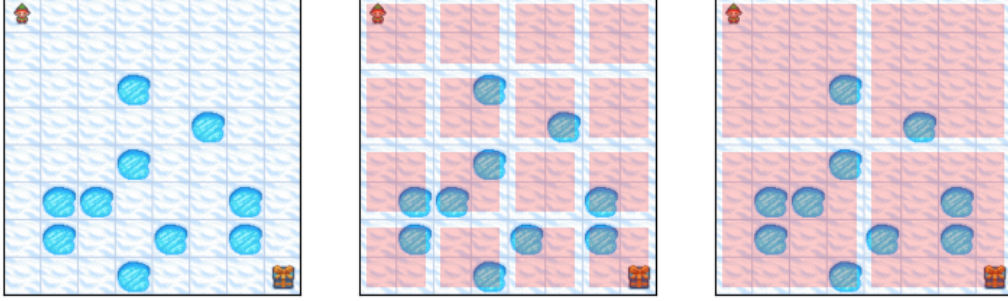


Figure 8.3: Abstraction functions applied to the Frozen Lake environment

grid layout remains fixed across episodes, but the agents and goals starting positions vary.

To apply MANGO, we introduce a hierarchy of abstraction functions (ϕ_i) that compress contiguous regions of the grid into supercells. At abstraction level (i), each $((2^i \times 2^i))$ -block of the original grid is collapsed into a single abstract cell, while maintaining one-hot encodings of the agents and goals positions. By construction, these abstractions preserve connectivity and consistent transitions, satisfying the criteria outlined in Section 8.3.

At each abstraction level (i), the agent has access to movement actions in the compressed grid: **UP**, **LEFT**, **DOWN**, **RIGHT**. Executing an abstract action corresponds to moving between adjacent abstract cells at level (i). Environmental dynamics such as collisions with walls or falling into holes are naturally incorporated into the transition structure of that abstraction layer.

Figure 8.3 illustrates how blocks of the map are pooled at each layer, visually conveying which states are aggregated and how transitions map between adjacent abstract states.

A set of parallel workers collects rollouts by interacting with the environment, producing trajectories of states, rewards, and terminations that serve as input for off-policy RL updates.

Training proceeds bottom-up through the hierarchy. The first (lowest) layer of MANGO trains for a prescribed number of steps, learning a policy that captures basic navigation skills such as moving efficiently and avoiding holes. Once this budget is exhausted (or performance meets a desired threshold) the layers parameters are frozen. From that point onward, higher layers can treat it as part of the environments transition model, delegating fine-

grained control to the frozen sub-policy. Each successive layer then operates on increasingly abstract representations of the map, training its own policy while relying on the capabilities of the levels below. This process continues until all layers have been trained.

We experimented with several objective functions to balance early improvements against final convergence quality. These included maximizing the area under the reward-step curve, combining final accuracy with training duration, and rewarding both rapid and stable convergence. Results highlight the methods sensitivity to early training: insufficient training at the first layer causes errors to cascade upward, limiting overall performance. Conversely, overly aggressive learning rates can yield fast short-term gains but hinder fine-tuning at later stages.

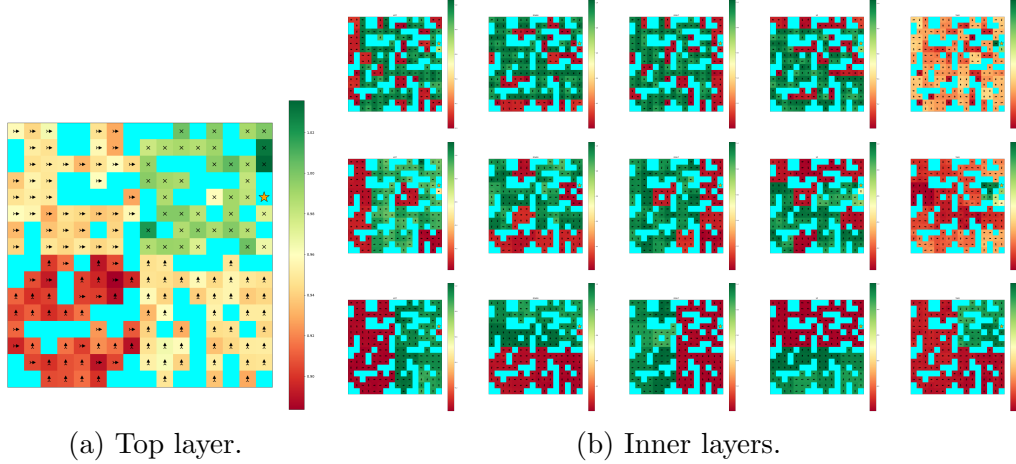


Figure 8.4: Comparison of task-level Q-values (a) and hierarchical Q-values across abstraction levels (b) illustrating how the trained agent’s behavior is guided across different positions within the environment.

The hierarchical training strategy proves effective on 16×16 randomly generated Frozen Lake maps, provided hyperparameters are tuned with care. In many trials, the agent achieves near-perfect completion rates once each layer’s policy is allocated sufficient training steps to master its corresponding abstraction. However, even minor misconfigurations, such as inappropriate learning rates, suboptimal soft-update schedules, or poorly chosen layer-wise training budgets, can lead to partial convergence or outright failure.

A consistent pattern emerges: when the base layer fails to adequately

learn local navigation and hole avoidance, upper layers inherit a brittle foundation. This unreliability propagates upward, preventing the stacked policy from reaching optimal performance. Conversely, when the first layer converges robustly, higher layers can leverage its reliable transitions to compose longer, more abstract paths, accelerating both exploration and convergence. Yet, this hierarchical advantage is fragile: overly aggressive learning rates may produce fast initial gains at the lower levels but undermine fine-tuning, while pruning heuristics such as those in Optuna can prematurely terminate runs that, despite slower starts, might have ultimately converged successfully.

8.8 Conclusion

This work presented MANGO, a hierarchical reinforcement learning framework that decomposes complex environments into layers of abstract state transitions and nested options. By structuring decision-making across multiple levels of abstraction, MANGO improving sample efficiency and promoting an interpretable policy structure where each decision can be directly tracked back to the implementation of a particular local skill or as a small step of a more complex long term plan.

Experiments in sparse-reward grid environments highlight the frameworks effectiveness when appropriately tuned, though results remain sensitive to the reliability of lower-layer policies and the specifics of training schedules. Because higher-level behavior depends on the reliability of lower-level options, insufficient training or poorly formed abstractions at early layers propagate upward, amplifying minor deviations into substantial errors. Our experiments illustrate this clearly. When the foundational layer struggled to learn accurate micro-transitions, the layers above it learned distorted value landscapes. The benefits of abstraction are fully realized only when the lower layers provide stable, predictable scaffolds on which to build on.

An important aspect to bring to attention is the design of the abstractions themselves. In the present work, the hierarchy of concepts and options was hand-crafted based on the spatial regularities of Frozen Lake, which allowed us to cleanly separate local navigation skills from more global reasoning. But environments without such clear structure pose a more challenging design. A system like MANGO ultimately succeeds or struggles based on how well its abstractions partition the environment into units that are both learnable and behaviorally meaningful.

Finally, the results suggest several opportunities for extending the frame-

work. While the current implementation trains layers sequentially, end-to-end optimization could mitigate some of the propagation issues that arise when early layers underperform. More ambitiously, the hierarchy itself could be learned rather than imposed, using methods from representation learning or option discovery. These extensions aim to enhance MANGOs scalability, robustness, and autonomy, enabling its deployment in increasingly complex real-world domains.

Part III

Encoding Desirable Properties into Models

9

Fair Representation Learning

9.1 Introduction

Fair classification has emerged as a critical challenge, driven both by international regulations and the increasing reliance on automated systems in high-stakes decision-making. Existing approaches often rely on adversarial learning or distribution matching across sensitive groups; however, adversarial learning can be unstable, and distribution matching tends to be computationally expensive.

To address these limitations, we propose a novel method based on the characteristic function distance. Our approach ensures that the learned representation contains minimal sensitive information while maintaining strong predictive performance for downstream tasks. By leveraging characteristic functions, we achieve a more stable and efficient solution compared to traditional approaches. Furthermore, we introduce a simple relaxation of the objective function that guarantees fairness in common classification models without degrading performance. Experimental results on benchmark datasets show that our method consistently matches or surpasses existing techniques in terms of both fairness and predictive accuracy. Importantly, it also maintains robustness and computational efficiency, making it a practical choice for real-world applications.

Algorithmic fairness has become a central concern in deploying automated decision-making systems, particularly in sensitive domains such as hiring, lending, criminal justice, and healthcare [Bla24, WZZ22, GMU24, FBD⁺24]. The growing reliance on these systems has intensified concerns that they may reinforce or amplify existing societal biases [BHN23, DCD, VTM21, BSS⁺24]. In response, a substantial body of research has focused on detecting, analyzing, and mitigating bias throughout the algorithmic pipeline [SHG⁺23, ZWS⁺13, MOW19, CLC⁺24].

Numerous fairness definitions and metrics have been proposed, reflecting diverse normative and technical perspectives [DHP⁺12, HPS16, DYZH20]. A common principle is statistical independence, which requires that model predictions be independent of sensitive attributes such as gender or ethnicity [R21, BHN23]. This principle is closely related to demographic parity, which demands that the probability of receiving a favorable outcome be approximately equal across demographic groups defined by the sensitive attribute [BHN23].

A simple strategy for enforcing demographic parity is to exclude the sensitive attribute from the input, commonly referred to as Fairness through Unawareness [CAB⁺23]. However, this approach is often ineffective in practice, since sensitive information may still be captured indirectly through correlated features (so-called proxy variables) [PSK22]. Eliminating all proxies typically degrades predictive accuracy, as task-relevant information is also lost [CAB⁺23].

To better balance fairness and predictive accuracy, many methods aim to learn fair representations by projecting input data into a latent space that obscures sensitive information while preserving task-relevant structure [ZWS⁺13, MCPZ18, MGB⁺18, BRV22]. These approaches typically fall into two categories: specialized and general. Specialized representations are tailored to a specific task, filtering out sensitive and irrelevant features for improved predictive performance. General representations, in contrast, are task-agnostic and designed for broad applicability across tasks, but are harder to learn and may compromise predictive power.

A popular class of methods employs Variational Autoencoders (VAEs) to learn fair, specialized representations [LSL⁺17, MGB⁺18]. In these methods, fairness is optimized indirectly through a variational bound or via Maximum Mean Discrepancy (MMD). However, both strategies may misalign with fairness objectives, often resulting in suboptimal outcomes [BRV22].

Other methods rely on adversarial networks, which enforce specialized fair

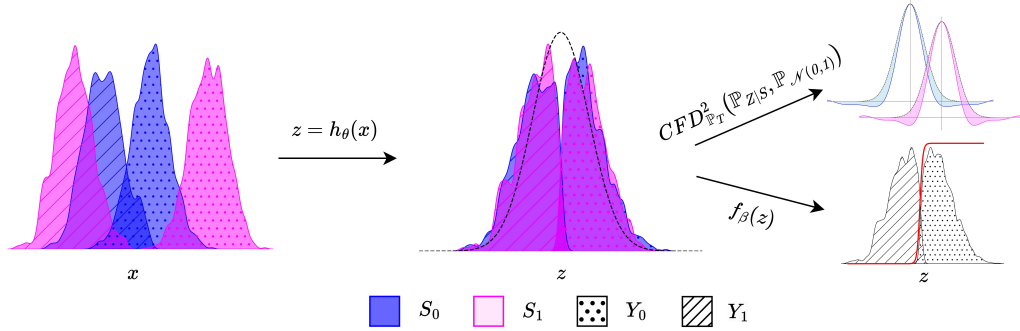


Figure 9.1: Overview of the proposed approach. Each sample X is associated with a sensitive attribute S and a target label Y . The conditional distribution $\mathbb{P}_{X|S}$ is mapped to a new distribution $\mathbb{P}_{Z|S}$, which is encouraged to resemble a Gaussian Distribution via the Characteristic Function Distance (CFD). The encoded representation Z minimizes $\Delta(\mathbb{P}_{Z|S_0}, \mathbb{P}_{Z|S_1})$ while retaining task-relevant information.

representations through a minimax game between a predictor and a fairness discriminator [ES15, JMVS⁺20]. While effective in some settings, these approaches are often unstable and highly sensitive to hyperparameters, making it difficult to manage the fairness-performance trade-off [EG18]. Moreover, fairness guarantees are tied to the specific adversary used during training and may fail if a stronger or different adversary is introduced at test time [MGB⁺18, XZS⁺20, GFDVS21, SS19].

Fair Normalizing Flows (FNF) [BRV22] have emerged as a state-of-the-art alternative, offering representations that are provably robust to any adversary. However, FNF also comes with significant limitations. It requires access to sensitive attributes at inference time, which conflicts with privacy considerations. It further demands training a separate model for each sensitive group, making it computationally costly, particularly for multi-class attributes or large datasets. Finally, while FNF is designed to learn general representations, the inclusion of a classification loss biases the model toward task-specific features. This task alignment risks embedding group-specific information into the representation, potentially undermining fairness guarantees through representation bias.

These issues highlight a fundamental tension between general and specialized representations. General representations offer broad transferability across tasks but often sacrifice predictive performance. Specialized repre-

sentations, by contrast, achieve stronger accuracy by aligning closely with a specific prediction task. Even approaches like FNF, which explicitly pursue generalization, ultimately rely on task-specific loss functions to achieve competitive accuracy, suggesting that some degree of specialization is not merely incidental but necessary for strong predictive performance. Recognizing this, specialization should be viewed as not only pragmatic but essential.

Adversarial methods embody this principle by explicitly tailoring representations to task-specific objectives, yet they suffer from unstable training dynamics and weaker fairness guarantees [BRV22, MGB⁺18, SS19]. This motivates the need for alternative approaches that preserve the benefits of specialization while providing stronger and more transparent fairness guarantees.

To this end, we propose a novel framework that embraces specialization in representations while rigorously enforcing fairness, without relying on adversarial training. Our method not only provides strong theoretical guarantees but is also simpler and more computationally lightweight than existing alternatives. This reinforces the case for specialized representations: in many real-world settings, the advantages of transferability from general representations are outweighed by the higher accuracy and efficiency achievable when models are specialized and retrained as needed. Our key contributions are as follows:

- We propose a novel approach to distribution matching based on characteristic functions. This formulation eliminates the need for auxiliary components such as variational autoencoders, adversarial networks, or normalizing flows.
- We derive a simplified variant of the framework tailored to classification, which provides formal guarantees on the amount of sensitive information accessible to downstream classifiers.
- We conduct extensive experiments showing that our method effectively removes sensitive information while achieving state-of-the-art accuracy and producing significantly fairer representations.

In summary, our approach offers a principled alternative for learning fair, specialized representations. It combines simplicity, training stability, and strong fairness guarantees, all without requiring access to sensitive attributes at inference. This makes the method both practical and privacy-preserving, delivering promising improvements over the current state of the art [BRV22].

9.2 Related Work

In the context of fairness through representation learning, Variational Autoencoders (VAEs) have been widely explored to disentangle sensitive attributes from learned data representations [LSL⁺17, MGB⁺18, CMJ⁺19, LAR⁺19]. In this setting, the encoder extracts task-relevant representations while the decoder reconstructs the input. Although reconstruction may retain extra information, the optimization process encourages representations tailored to the prediction task.

9.2.1 Variational Autoencoders

A prominent example is the Variational Fair Autoencoder (VFAE) [LSL⁺17], which extends the standard VAE framework by enforcing invariance of the latent space with respect to protected attributes. This is achieved by introducing a penalty term that explicitly encourages independence between latent representations and sensitive attributes. A related approach by Moyer et al. [MGB⁺18] combines VAEs with the Variational Information Bottleneck (VIB) to learn representations that remain both informative for prediction and robust to variations in sensitive inputs.

9.2.2 Adversarial Training

Another line of research addresses fairness-aware learning through adversarial and information-theoretic methods designed to induce invariant representations [XDD⁺17, RB19, SKG⁺19]. In these approaches, a model, typically an encoder or predictor, is trained in opposition to an adversary that seeks to recover sensitive attributes from the learned representation. The adversarial objective pressures the model to produce latent features that obscure sensitive information, thereby promoting invariance with respect to protected variables [ES15, FYL⁺19].

Xie et al. [XDD⁺17] proposed a general adversarial framework for learning representations invariant to arbitrary nuisance attributes. Their method formulates the process as a three-player minimax game involving an encoder, a task-specific predictor, and a discriminator that attempts to infer the nuisance attribute. Madras et al. [MCPZ18] introduced Learning Adversarially Fair and Transferable Representations (LAFTR), which integrates adversarial objectives aligned with specific fairness definitions to ensure that representations remain fair even when deployed in downstream classifiers without explicit fairness constraints.

Roy and Bodetti [RB19] extended this paradigm with MaxEnt-ARL, which maximizes the entropy of the adversary’s prediction of the sensitive attribute rather than minimizing its accuracy. This strategy enhances privacy by removing the requirement for sensitive labels during encoder training. Jaiswal et al. [JMVS⁺20] proposed Adversarial Forgetting, a framework that separates the learning of expressive representations from the selective removal of sensitive or nuisance information via a dedicated forget-gate mechanism. Building on these ideas, FR-Train [RLWS20] introduced a mutual information-based formulation and incorporated a second adversary to improve robustness against poisoned data.

9.2.3 Other Approaches

Recognizing the limitations of adversarial frameworks, most notably their instability and lack of formal guarantees [EG18, GFDVS21, FYL⁺19], researchers have investigated alternative paradigms for learning fair representations. Jiang et al. [JPS⁺20] proposed a theoretically grounded approach that enforces demographic parity by minimizing the Wasserstein-1 distance between model outputs across sensitive groups. Tucker and Shah [TS22] introduced Concept Subspace Networks (CSNs), a prototype-based architecture that unifies fair and hierarchical classification within a single model.

Building on these developments, Balunovi et al. [BRV22] proposed Fair Normalizing Flows (FNF), a framework that provides provable fairness guarantees against any downstream adversary. FNF currently represents the state of the art in fair representation learning. It employs separate normalizing flow encoders for each sensitive group and enforces fairness by minimizing the statistical distance between the resulting latent distributions. This formulation enables exact likelihood computation in the latent space, making it possible to derive theoretical upper bounds on unfairness for any downstream classifier, a property rarely achieved in prior methods.

Furthermore, Balunovi et al. [BRV22] demonstrate that adversarial learning often induces a false sense of fairness [FYL⁺19, XZS⁺20, EG18, GFDVS21]. In particular, they show that adversarially learned representations frequently fail when evaluated against more powerful families of classifiers. While adversarial learning remains the only approach designed to produce specialized representations, this evidence underscores the urgent need for new methods that achieve specialization without inheriting the instability and fragility of adversarial training.

9.3 Background

Let $X \in \mathbb{R}^d$ denote a feature vector, $Y \in 0, 1$ a binary label, and $S \in s_1, \dots, s_n$ a sensitive attribute, drawn from a joint distribution $\mathbb{P}_{X,Y,S}$. In fairness-critical applications, the most common setting is a *binary* sensitive attribute $S \in \{0, 1\}$.

Traditional classification algorithms fit a classifier $f_\theta : \mathbb{R}^d \rightarrow 0, 1$ to predict the task label Y from X . However, since the sensitive attribute S is often statistically correlated with both the features X and the target Y , fairness concerns naturally arise. A prevalent strategy in state-of-the-art fairness-aware methods is to construct fair representations Z from X prior to predicting Y .

9.3.1 Statistical Distance and Adversarial Evaluation

Strict fairness constraints often introduce an accuracy-fairness trade-off, as correlations between S and Y mean that suppressing sensitive information can reduce predictive performance or, conversely, allow leakage of S into the classification. To manage this tension, we adopt the notion of ϵ -*fairness*, where ϵ denotes the statistical distance between the learned representations $\mathbb{P}_{Z|S}$. This notion connects to bounds derived by Madras et al. [MCPZ18], which relate statistical distances to several fairness metrics (see section 9.3.2).

In particular, they consider the Total Variation (TV) distance, defined for two distributions $\mathbb{P}_0$ and $\mathbb{P}_1$ as:

$$\Delta(\mathbb{P}_0, \mathbb{P}_1) = \frac{1}{2} \int_{\mathcal{X}} |\mathbb{P}_0(x) - \mathbb{P}_1(x)| dx. \quad (9.1)$$

Balunovi et al. [BRV22] estimate the statistical distance directly. However, this can be extremely challenging in the general case, and inaccuracies in estimation may lead to unreliable fairness guarantees. Since our proposed method does not impose specific distributional assumptions on Z , we instead adopt a uniform strategy of *adversarial evaluation* to quantify fairness throughout this work.

For an optimal adversarial classifier f , the statistical distance between conditional distributions can be expressed precisely as:

$$\sup_{\mathcal{F}} \max_{f \in \mathcal{F}} \mathbb{P}(Y = f(X)) = \frac{1}{2} (1 + \Delta(\mathbb{P}_{Z|s=0}, \mathbb{P}_{Z|s=1})). \quad (9.2)$$

It is important to note that *adversarial evaluation* is conceptually distinct from *adversarial learning*. Adversarial learning aims to learn a fair

latent representation Z by optimizing eq. (9.2) directly through a min-max game, which often introduces training instability [FYL⁺19, MGB⁺18, EG18]. Moreover, adversarial frameworks are typically robust only against the specific class of functions used during training. In contrast, adversarial evaluation merely assesses the fairness of a fixed representation Z by measuring how accurately a classifier (commonly a multilayer perceptron) can recover the sensitive attribute S from Z .

9.3.2 Additional Fairness Metrics

Although various fairness criteria have been proposed, balancing the fairness-accuracy trade-off remains challenging.

Demographic Parity (DP) is a widely used group fairness criterion that requires the outcome of a decision-making process to be independent of a sensitive attribute such as race, gender, or age [DHP⁺12, DEHH21]. Let h_θ denote a representation function that maps inputs X to a latent space $Z = h_\theta(X)$, and let f_θ denote a classifier applied to this representation, producing predictions $\hat{Y} = f_\theta(h_\theta(X))$. Then, DP requires the probability of receiving a favorable outcome (e.g., $\hat{Y} = 1$) to be the same across all demographic groups $s \in \mathcal{S}$:

$$\mathbb{P}(\hat{Y} = 1 \mid S = s) = \mathbb{P}(\hat{Y} = 1) \quad \forall s \in \mathcal{S}. \quad (9.3)$$

This states that the positive outcome rate for any group s should be equal to the overall positive outcome rate in the population. Since demographic parity is independent of the ground truth labels, it is especially salient in contexts where reliable ground truth information is hard to obtain and a positive outcome is desirable, including employment, credit, and criminal justice [DYZH20, GP18].

To quantify deviations from this ideal, we measure the Demographic Parity Difference (DPD). In binary group settings (e.g., involving an advantaged group a and a disadvantaged group d), it is defined as:

$$\text{DPD} = \mathbb{P}(\hat{Y} = 1 \mid S = a) - \mathbb{P}(\hat{Y} = 1 \mid S = d). \quad (9.4)$$

In real-world scenarios, sensitive attributes are often multi-class (e.g., race with more than two categories). In these cases, generalizations for measuring DPD violation include:

- The *Maximum Pairwise Difference*, capturing the largest disparity in positive outcome rates between any two groups:

$$\text{DPD} = \max_{s_i, s_j \in \mathcal{S}} \left| \mathbb{P}(\hat{Y} = 1 \mid S = s_i) - \mathbb{P}(\hat{Y} = 1 \mid S = s_j) \right|. \quad (9.5)$$

- The *Average Absolute Pairwise Difference*, computing the average disparity across all pairs:

$$\text{DPD} = \frac{1}{|\mathcal{S}|^2} \sum_{s_i \in \mathcal{S}} \sum_{s_j \in \mathcal{S}} \left| \mathbb{P}(\hat{Y} = 1 \mid S = s_i) - \mathbb{P}(\hat{Y} = 1 \mid S = s_j) \right|. \quad (9.6)$$

Since achieving perfect demographic parity (where all the above differences are zero) can be impractical, the objective is often relaxed to minimizing the Demographic Parity Distance:

$$\Delta^{\text{DP}} = \sum_{s \in \mathcal{S}} \left| \mathbb{P}(\hat{Y} = 1 \mid S = s) - \mathbb{P}(\hat{Y} = 1) \right|. \quad (9.7)$$

While demographic parity is often defined in terms of these outcome rate differences, it can also be understood more generally in terms of statistical distance between group-conditional distributions of model predictions or representations [MCPZ18]. Let Z_0 and Z_1 denote the distributions of these representations conditioned on different sensitive attribute values $S = 0$ and $S = 1$, respectively. Then, for any measurable test function f_θ applied to the model outputs, one can define the test discrepancy as:

$$\Delta^{\text{DP}}(f_\theta \circ h_\theta) = \|\mathbb{E}_{Z_0}[f_\theta(Z)] - \mathbb{E}_{Z_1}[f_\theta(Z)]\|. \quad (9.8)$$

The statistical distance (or total variation distance) between Z_0 and Z_1 is the supremum of this discrepancy over all measurable test functions μ :

$$\Delta^*(Z_0, Z_1) = \sup_{\mu} \|\mathbb{E}_{Z_0}[\mu(Z)] - \mathbb{E}_{Z_1}[\mu(Z)]\|. \quad (9.9)$$

It follows that for any f_θ ,

$$\Delta^{\text{DP}}(f_\theta \circ h_\theta) \leq \Delta^*(Z_0, Z_1), \quad (9.10)$$

and $\Delta^{\text{DP}}(f_\theta \circ h_\theta) = 0$ if and only if $f_\theta(h_\theta(X)) \perp S$, i.e., demographic parity holds.

In contrast to Demographic Parity, the Equal Opportunity (EO) criterion incorporates information about the target variable Y [HPS16]. Specifically, EO requires that the true positive rate (TPR) be equal across all demographic groups. Formally:

$$\mathbb{P}(\hat{Y} = 1 \mid S = s, Y = 1) = \mathbb{P}(\hat{Y} = 1 \mid Y = 1) \quad \forall s \in \mathcal{S}. \quad (9.11)$$

This fairness criterion is especially salient in domains where false negatives are particularly harmful and reasonably accurate ground truth labels are available, such as in healthcare, criminal justice, and risk assessment [Cho17, SKLM⁺20, OPVM19]. The corresponding EO distance is defined as:

$$\Delta^{\text{EO}}(f_\theta \circ h_\theta) = \|\mathbb{E}_Z[f_\theta(Z) \mid S = 0, Y = 1] - \mathbb{E}_Z[f_\theta(Z) \mid S = 1, Y = 1]\| \quad (9.12)$$

which can be upper bounded by the objective value of an optimal adversary g^* trained to distinguish between $Z \mid Y = 1$ for different groups [MCPZ18].

An alternative fairness criterion is Equalized Odds (EOd) [HPS16], which extends Equal Opportunity by requiring that prediction outcomes be conditionally independent of the sensitive attribute given the true label. Formally, EOd demands that for all $s \in \mathcal{S}$:

$$\mathbb{P}(\hat{Y} = 1 \mid Y = 1, S = s) = \mathbb{P}(\hat{Y} = 1 \mid Y = 1) \quad \forall s \in \mathcal{S} \quad (9.13)$$

and

$$\mathbb{P}(\hat{Y} = 1 \mid Y = 0, S = s) = \mathbb{P}(\hat{Y} = 1 \mid Y = 0) \quad \forall s \in \mathcal{S} \quad (9.14)$$

These constraints ensure that both the true positive rate (TPR) and false positive rate (FPR) are equal across all groups [ZT24, GKP19]. This criterion is particularly applicable in decision-making contexts where reliable ground truth labels are available and both false negatives and false positives incur significant societal or personal costs [DK25, YCM24]. The EOd distance is defined as:

$$\begin{aligned} \Delta^{\text{EOd}}(f_\theta \circ h_\theta) = & \|\mathbb{E}_Z[f_\theta(Z) \mid S = 0, Y = 1] - \mathbb{E}_Z[f_\theta(Z) \mid S = 1, Y = 1]\| \\ & + \|\mathbb{E}_Z[f_\theta(Z) \mid S = 0, Y = 0] - \mathbb{E}_Z[f_\theta(Z) \mid S = 1, Y = 0]\|. \end{aligned} \quad (9.15)$$

9.3.3 Logistic Regression

Logistic Regression (LR) is a popular classification algorithm used to model the probability of a binary outcome $Y \in \{0, 1\}$ as a function of a set of predictors $X \in \mathbb{R}^d$. The model is defined by the logistic function $\sigma_\beta : \mathbb{R}^d \rightarrow (0, 1)$:

$$\sigma_\beta(x) = \frac{1}{1 + e^{-\beta_{[0]} - \sum_{i=1}^d \beta_{[i]} x_{[i]}}}, \quad (9.16)$$

where the coefficients $\beta \in \mathbb{R}^{d+1}$ are the model parameters, with $\beta_{[0]}$ denoting the bias term. The model output $\sigma_\beta(x)$ is used to learn $\mathbb{P}(Y = 1 \mid X = x)$ via maximum likelihood estimation. To achieve this, the objective function to be minimized is given by the negative log-likelihood:

$$\mathcal{L}^{\text{LR}}(\beta) = -\mathbb{E}_{X,Y} [Y \log(\sigma_\beta(X)) + (1 - Y) \log(1 - \sigma_\beta(X))] \quad (9.17)$$

A key advantage of LR over other families of classifiers, such as MLP, is the convexity of its objective function. Since this loss function is convex, it has a global optimum that can be provably reached. This property not only guarantees convergence but also enables the use of second-order optimization techniques such as Newton-Raphson or its variant, Iteratively Reweighted Least Squares. These methods leverage the Hessian of the loss function to achieve quadratic convergence, significantly accelerating the optimization compared to first-order methods like gradient descent.

9.4 Fair Representations matching Characteristic Function

As discussed in section 9.3.1, to incentivize fair representation, we need to match the different conditional distributions $\mathbb{P}_{Z|S}$, thus minimizing their relative statistical distance. To achieve this, we propose the addition of a differentiable penalty term based on the Characteristic Function Distance. We refer to the overall approach as Fairness matching Characteristic Function (FmCF).

9.4.1 Characteristic Function Distance

Given a random variable $X \in \mathbb{R}^d$, an alternative way to describe the distribution of X is the Characteristic Function (CF) $\varphi_X : \mathbb{R}^n \rightarrow \mathbb{C}$. Denoting by $\mathbb{P}_X$ the probability measure of X , the characteristic function φ_X is defined

as:

$$\varphi_X(t) = \mathbb{E}_X [e^{i\langle X, t \rangle}] = \int_{x \in \mathbb{R}^d} e^{i\langle x, t \rangle} d\mathbb{P}_X. \quad (9.18)$$

The CF is closely related to the notion of Fourier Transform, and inherits several important properties. It always exists, it's bounded $|\varphi_X(t)| \leq 1$, but most importantly, there is a one-to-one correspondence between probability measures and CFs [Luk70]. That is, given two random variables X and Y , with probability measures $\mathbb{P}_X$ and $\mathbb{P}_Y$ then it holds that:

$$\varphi_X = \varphi_Y \iff \mathbb{P}_X = \mathbb{P}_Y. \quad (9.19)$$

The CF can be used to define a distance metric between probability measures [ASS20]. Given two random variables $X \in \mathbb{R}^d$, $Y \in \mathbb{R}^d$ with probability measures $\mathbb{P}_X$ and $\mathbb{P}_Y$, the CFD is defined as:

$$\text{CFD}_{\mathbb{P}_T}^2(\mathbb{P}_X, \mathbb{P}_Y) = \mathbb{E}_T [|\varphi_X(T) - \varphi_Y(T)|^2]. \quad (9.20)$$

The value of $\text{CFD}_{\mathbb{P}_T}$ depends on the weighting kernel $\mathbb{P}_T$. When $\text{supp}(\mathbb{P}_T) = \mathbb{R}^d$, then the CFD is strict, in the sense that $\text{CFD}_{\mathbb{P}_T}(\mathbb{P}_X, \mathbb{P}_Y) = 0 \iff \mathbb{P}_X = \mathbb{P}_Y$. Common choices for $\mathbb{P}_T$ that ensure this property are the Normal and Laplace distributions.

Instead of matching $\mathbb{P}_{Z|S}$ to each other, each $\mathbb{P}_{Z|S}$ is matched independently to a common target distribution. This simplifies and stabilizes the training procedure.

In this work, we use a standard Normal distribution as the target, although in principle, any distribution could be used. The CF of the standard Normal distribution $\varphi_{\mathcal{N}}$ is given by:

$$\varphi_{\mathcal{N}}(t) = \int_{x \in \mathbb{R}^d} (2\pi)^{-\frac{d}{2}} e^{-\frac{\|x\|^2}{2}} e^{i\langle x, t \rangle} dx = e^{-\frac{\|t\|^2}{2}}. \quad (9.21)$$

The CFD between $\mathbb{P}_{Z|S}$ and the target distribution can be estimated using Monte Carlo sampling. Given a batch of independent identically distributed samples $z_1, \dots, z_n$ taken from $\mathbb{P}_{Z|S}$ and $t_1, \dots, t_j$ taken from the weighting kernel $\mathbb{P}_T$ the CFD can be approximated as:

$$\text{CFD}_{\mathbb{P}_T}^2(\mathbb{P}_{Z|S}, \mathbb{P}_{\mathcal{N}}) \approx \frac{1}{k} \sum_{j=1}^k |\varphi_{\mathcal{N}}(t_j) - \hat{\varphi}_{Z|S}(t_j)|^2. \quad (9.22)$$

where $\hat{\varphi}_{Z|S}(t) = \frac{1}{n} \sum_{i=1}^n e^{i\langle t, z_i \rangle}$ denotes the empirical CF. The total penalty term is obtained by summing the CFD estimate across all different sensitive groups $\mathcal{S}$

$$\mathcal{L}^{\text{CF}} = \sum_{s \in \mathcal{S}} \frac{1}{k} \sum_{j=1}^k |\varphi_{\mathcal{N}}(t_j) - \hat{\varphi}_{Z|S=s}(t_j)|^2. \quad (9.23)$$

9.4.2 Comparison to Adversarial Learning and FNF

Similarly to FmCF, adversarial learning produces highly specialized representations. However, such methods have been shown to provide only an illusory sense of fairness [MGB⁺18, XZS⁺20, SS19], largely due to the inherent limitations of the underlying optimization problem. In contrast, our proposed approach learns representations that remain strongly predictive of Y while minimizing information about S , without relying on adversarial losses.

Compared to Fair Normalizing Flows (FNF) [BRV22], FmCF offers several key advantages over normalizing flow-based methods. First, normalizing flows (NFs) require carefully designed architectures: only specific layer structures permit efficient computation of the Jacobian determinant, and invertibility constraints mandate that the dimensionality of the input vector X be preserved throughout the transformation. These restrictions make NFs computationally expensive, especially for high-dimensional data. In contrast, FmCF allows any function approximator to serve as the encoder. This architectural flexibility enables the learned representation Z to capture only the information relevant for the downstream task (e.g., classification).

Second, FmCF eliminates the need for sensitive information during deployment. Since the classifier debiases X internally, it does not require access to the sensitive attribute S at test time. FNF, by contrast, not only requires S during inference but also trains a separate normalizing flow for each sensitive group, substantially increasing computational cost. This distinction is particularly important for reducing potential discrimination and mitigating concerns about the explicit collection of sensitive data.

9.5 Fair Classification matching Sufficient Statistics

The method introduced in section 9.4 provides a versatile framework for learning specialized representations Z from X independent from S for a general task. However, in practice, most fairness-aware scenarios involve classi-

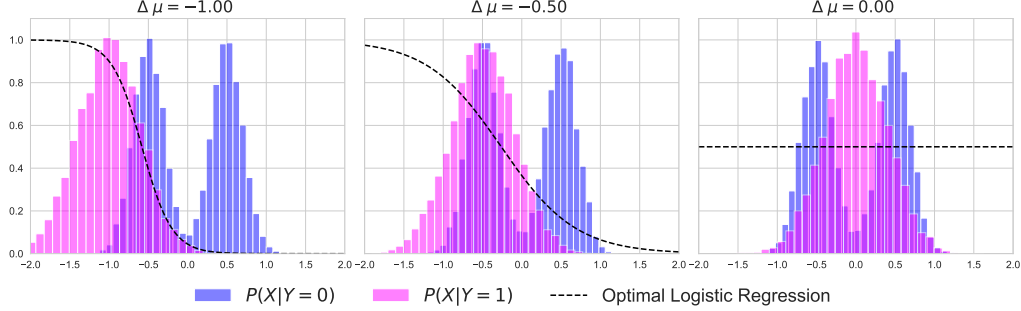


Figure 9.2: The closer $\mathbb{E}[X|y=0]$ is to $\mathbb{E}[X|y=1]$, the less predictive power the feature has, causing the LR coefficient β to approach 0.

fication [WZW18, MW20, ZVRG17]. By simplifying the approach, tailoring it to fair classification, we can provide provable post-hoc fairness guarantees. Furthermore, we can also make it computationally cheaper, relaxing the need for a Monte Carlo estimate of the CF.

9.5.1 Connection to Distribution Moments

The penalty term introduced in section 9.4 requires sampling multiple points $t \sim \mathbb{P}_T$ from the weighting kernel, at which the empirical CF is evaluated. However, the number of sampled points needed to obtain a reliable estimate of the CFD grows rapidly with the dimensionality of the data [ASS20].

To address this issue, we propose an alternative that reduces the number of evaluation points required. Specifically, we consider the multidimensional Maclaurin expansion of the CF:

$$\varphi_X(t) \approx \sum_{n=0}^{\infty} \sum_{\substack{n_1, \dots, n_d \\ \text{s.t. } \sum_i n_i = n}} \frac{t_{[1]}^{n_1} \cdots t_{[d]}^{n_d}}{n_1! \cdots n_d!} \left(\frac{\partial^n \varphi_X}{\partial t_{[1]}^{n_1} \cdots \partial t_{[d]}^{n_d}} \right) \Big|_{t=0}. \quad (9.24)$$

Assuming that φ_X is analytic, fairness could in principle be ensured by matching its derivatives evaluated at $t = 0$, thereby removing the need to sample from $\mathbb{P}_T$. While statistical tools exist to estimate the full series (section 9.7), in practice they often introduce severe training instabilities.

A more pragmatic solution is to truncate the expansion at some finite order N . Although this objective does not guarantee exact equality of the distributions, we demonstrate that truncation at $N = 2$ already suffices to ensure fair classification.

This observation follows from the direct connection between the derivatives of the CF and the moments of the underlying distribution.

Theorem 9.5.1. *The n -derivative of the CF evaluated at $t = 0$ is related to the n -th moment of the distribution:*

$$\left. \frac{\partial^n \varphi_X}{\partial t_{[k_1]}, \dots, \partial t_{[k_n]}} \right|_{t=0} = i^n \mathbb{E}_X [X_{[k_1]} \dots X_{[k_n]}]. \quad (9.25)$$

Proof. Taking the derivative inside the expectation

$$\frac{\partial^n \varphi_X}{\partial t_{[k_1]}, \dots, \partial t_{[k_n]}} = \frac{\partial^n \mathbb{E}_X [e^{i\langle X, t \rangle}]}{\partial t_{[k_1]}, \dots, \partial t_{[k_n]}} \quad (9.26)$$

$$= \mathbb{E}_X \left[\frac{\partial^n e^{i\langle X, t \rangle}}{\partial t_{[k_1]}, \dots, \partial t_{[k_n]}} \right] \quad (9.27)$$

$$= \mathbb{E}_X [i^n X_{[k_1]} \dots X_{[k_n]} e^{i\langle X, t \rangle}]. \quad (9.28)$$

Evaluating both sides at $t = 0$ concludes the proof: $\square$

Similarly, the n -th empirical moment is equivalent to the n -th derivative of the empirical CF.

9.5.2 Simplified Penalty for Classification

A generic Multi Layer Perceptron (MLP) with M layers used for classification can be decomposed into $M - 1$ encoder layers, $z = h_\theta(x)$, followed by a final logistic regression (LR) layer, $\hat{y} = \sigma_\beta(z)$. Applying a fairness penalty directly to z therefore allows us to restrict the analysis to the family of LR classifiers.

There exists an optimal condition under which LR is provably fair. Specifically, if the first moment satisfies $\mathbb{E}[Z | S]$ being identical for all $s \in S$, then the representation Z carries no predictive power for LR in inferring S . In this case, the optimal solution corresponds to $\beta = 0$.

Theorem 9.5.2. *For a representation $Z \in \mathbb{R}^d$ and a binary sensitive attribute $S \in \{0, 1\}$, the optimal LR classifier σ_{β^*} with S as target is invariant to $Z_{[i]}$ when $\mathbb{E}_{Z|S} [Z_{[i]}] = \mathbb{E}_Z [Z_{[i]}]$.*

Proof. Defining $x_{[0]} = 1$, the derivative of the logistic function can be expressed as

$$\frac{\partial \sigma_\beta(x)}{\partial \beta_{[i]}} = x_{[i]} \sigma_\beta(x) (1 - \sigma_\beta(x)). \quad (9.29)$$

Algorithm 1 FmCF loss term

```
1: Input: encoder  $h_\theta$ , decoder  $f_\theta$ , batch  $\mathcal{B} = \{(x_i, y_i, s_i) \sim \mathbb{P}_{X,Y,S}\}$ .
2:  $z \leftarrow h_\theta(x)$ 
3:  $\hat{y} \leftarrow f_\theta(z)$ 
4: for all  $s \in S$  do
5:   for all  $j \in \{1, \dots, k\}$  do
6:     Sample  $t_j \sim \mathbb{P}_T$ 
7:      $\varphi_{\mathcal{N}}(t_j) \leftarrow e^{-0.5\|t_j\|^2}$ 
8:      $\hat{\varphi}_{Z|s}(t_j) = \frac{1}{n} \sum_{i=1}^n e^{i\langle t_j, z_i \rangle}$ 
9:   end for
10: end for
11:  $\mathcal{L}_{\text{CF}} \leftarrow \sum_{s \in S} \frac{1}{k} \sum_{j=1}^k |\varphi_{\mathcal{N}}(t_j) - \hat{\varphi}_{Z|s}(t_j)|^2$ 
12:  $\mathcal{L} \leftarrow \mathcal{L}_C(\hat{y}, y) + \alpha \mathcal{L}_{\text{CF}}$ 
```

Algorithm 2 FmSS Training

```
1: Input: encoder  $h_\theta$ , linear regression  $f_\beta$ , batch  $\mathcal{B} = \{(x_i, y_i, s_i) \sim \mathbb{P}_{X,Y,S}\}$ .

2:  $z \leftarrow h_\theta(x)$ 
3:  $\hat{y} \leftarrow f_\beta(z)$ 
4: for all  $s \in S$  do
5:    $\sigma_s^2 \leftarrow \text{Var}[Z \mid S = s]$ 
6:    $\mu_s \leftarrow \mathbb{E}[Z \mid S = s]$ 
7: end for
8:  $\mathcal{L}_{\text{KL}} \leftarrow \sum_{s \in S} \|\sigma_s^2 + \mu_s^2 - 1 - \log \sigma_s^2\|_1$ 
9:  $\mathcal{L} \leftarrow \mathcal{L}_C(\hat{y}, y) + \alpha \mathcal{L}_{\text{KL}}$ 
```

The derivative of the log-likelihood function in eq. (9.17) can be expressed as

$$\frac{\partial \mathcal{L}^{\text{LR}}(\beta)}{\partial \beta_{[i]}} = -\frac{\partial}{\partial \beta_{[i]}} \mathbb{E}_{X,Y} [Y \log(\sigma_\beta(X)) + (1-Y) \log(1 - \sigma_\beta(X))] \quad (9.30)$$

$$= -\mathbb{E}_{X,Y} \left[Y \frac{1}{\sigma_\beta(X)} \frac{\partial \sigma_\beta(X)}{\partial \beta_{[i]}} - (1-Y) \frac{1}{(1 - \sigma_\beta(X))} \frac{\partial \sigma_\beta(X)}{\partial \beta_{[i]}} \right] \quad (9.31)$$

$$= -\mathbb{E}_{X,Y} [Y x_{[i]} (1 - \sigma_\beta(x)) - (1-Y) x_{[i]} \sigma_\beta(x)] \quad (9.32)$$

$$= -\mathbb{E}_{X,Y} [x_{[i]} (Y - \sigma_\beta(x))] . \quad (9.33)$$

At the optimum β^* , the gradient of the log-likelihood is zero. Considering the partial derivative with respect to $\beta_{[0]}$

$$0 = \frac{\partial \mathcal{L}^{\text{LR}}(\beta^*)}{\partial \beta_{[0]}} = -\mathbb{E}_{X,Y} [Y - \sigma_{\beta^*}(X)] \quad (9.34)$$

$$= -\mathbb{E}_Y [Y] + \mathbb{E}_X [\sigma_{\beta^*}(X)] \quad (9.35)$$

showing that $\mathbb{E}_Y [Y] = \mathbb{E}_X [\sigma_{\beta^*}(X)]$.

Considering the partial derivative with respect to $\beta_{[i]}$ for $i \neq 0$

$$0 = \frac{\partial \mathcal{L}^{\text{LR}}(\beta^*)}{\partial \beta_{[i]}} = -\mathbb{E}_{X,Y} [X_{[i]} (Y - \sigma_{\beta^*}(X))] \quad (9.36)$$

$$= \mathbb{E}_X [X_{[i]} \sigma_{\beta^*}(X)] - \mathbb{E}_Y [Y \mathbb{E}_{X|Y} [X_{[i]}]] . \quad (9.37)$$

Substituting $\mathbb{E}_X [\sigma_{\beta^*}(X)] = \mathbb{E}_Y [Y]$ and the hypothesis $\mathbb{E}_{X|Y} [X_{[i]}] = \mathbb{E}_X [X_{[i]}]$

$$0 = \frac{\partial \mathcal{L}^{\text{LR}}(\beta^*)}{\partial \beta_{[i]}} = \mathbb{E}_X [X_{[i]} \sigma_{\beta^*}(X)] - \mathbb{E}_Y [Y] \mathbb{E}_X [X_{[i]}] \quad (9.38)$$

$$= \mathbb{E}_X [X_{[i]} \sigma_{\beta^*}(X)] - \mathbb{E}_X [\sigma_{\beta^*}(X)] \mathbb{E}_X [X_{[i]}] \quad (9.39)$$

$$= \text{Cov} (X_{[i]}; \sigma_{\beta^*}(X)) \quad (9.40)$$

Unless $X_{[i]}$ is constant, the condition $\text{Cov} (X_{[i]}; \sigma_{\beta^*}(X)) = 0$ is obtained for $\beta_{[i]}^* = 0$, making $\sigma_{\beta^*}(X)$ independent of $X_{[i]}$. $\square$

Theorem 9.5.2 shows that, building an intermediate representation Z with $\mathbb{E}_{Z|S} [Z] = \mathbb{E}_Z [Z]$ is sufficient to ensure that LR models do not use information about the sensitive attribute S .

However, we can derive an even stronger guarantee, analyzing what happens for $\mathbb{E}_Z [Z_{[i]} | S] \approx \mathbb{E}_Z [Z_{[i]}]$. In particular, the mean and variance of $\mathbb{P}_{Z|S}$ are enough to fully characterize the predictive power of a feature.

Theorem 9.5.3. *The predictive power of a feature $Z_{[i]}$ goes to zero the more its conditional expectation is close to the marginal one. That is, given $\Omega = \frac{\|\mathbb{E}_{Z|S}[Z_{[i]}] - \mathbb{E}_Z[Z_{[i]}]\|^2}{\mathbb{E}_Z[Z_{[i]}^2]}$*

$$\lim_{\Omega \rightarrow 0} \beta_{[i]}^* = 0 \quad (9.41)$$

Proof. Following the same steps of the proof for theorem 9.5.2, we can show that $\mathbb{E}_X [\sigma_{\beta^*}(X)] = \mathbb{E}_Y [Y]$ and $\mathbb{E}_X [X_{[i]} \sigma_{\beta^*}(X)] = \mathbb{E}_Y [Y \mathbb{E}_{X|Y} [X_{[i]}]]$.

Therefore, the correlation $\text{Corr}(X_{[i]}; \sigma_{\beta^*}(X))$ can be expressed as:

$$\text{Corr}(X_{[i]}; \sigma_{\beta^*}(X)) = \frac{\mathbb{E}_X [(X_{[i]} - \mathbb{E}_X [X_{[i]}]) (\sigma_{\beta^*}(X) - \mathbb{E}_X [\sigma_{\beta^*}(X)])]}{\sqrt{\mathbb{E}_X [X_{[i]}^2] \mathbb{E}_X [\sigma_{\beta^*}^2(X)]}} \quad (9.42)$$

$$= \frac{\mathbb{E}_X [X_{[i]} \sigma_{\beta^*}(X)] - \mathbb{E}_X [X_{[i]}] \mathbb{E}_X [\sigma_{\beta^*}(X)]}{\sqrt{\mathbb{E}_X [X_{[i]}^2] \mathbb{E}_X [\sigma_{\beta^*}^2(X)]}}. \quad (9.43)$$

Using the triangle inequality $\mathbb{E}_X [\sigma_{\beta}(X)^2] \geq \mathbb{E}_X [\sigma_{\beta}(X)]^2$ and substituting $\sigma_{\beta}(X) = \mathbb{E}_Y [Y]$ and $\mathbb{E}_X [X_{[i]} \sigma_{\beta^*}(X)] = \mathbb{E}_Y [Y \mathbb{E}_{X|Y} [X_{[i]}]]$:

$$\text{Corr}(X_{[i]}; \sigma_{\beta^*}(X)) \leq \frac{\mathbb{E}_X [X_{[i]} \sigma_{\beta^*}(X)] - \mathbb{E}_X [X_{[i]}] \mathbb{E}_X [\sigma_{\beta^*}(X)]}{\sqrt{\mathbb{E}_X [X_{[i]}^2] \mathbb{E}_X [\sigma_{\beta^*}(X)]}} \quad (9.44)$$

$$= \frac{\mathbb{E}_Y [Y \mathbb{E}_{X|Y} [X_{[i]}]] - \mathbb{E}_X [X_{[i]}] \mathbb{E}_Y [Y]}{\sqrt{\mathbb{E}_X [X_{[i]}^2] \mathbb{E}_Y [Y]}} \quad (9.45)$$

$$= \frac{\mathbb{E}_Y [Y (\mathbb{E}_{X|Y} [X_{[i]}] - \mathbb{E}_X [X_{[i]}])]}{\sqrt{\mathbb{E}_X [X_{[i]}^2] \mathbb{E}_Y [Y]}}. \quad (9.46)$$

Assuming $\Omega = \frac{\|\mathbb{E}_{X|Y}[X_{[i]}] - \mathbb{E}_X[X_{[i]}]\|^2}{\mathbb{E}_X[X_{[i]}^2]} < \epsilon$ we can bound the correlation with

$$\text{Corr}(X_{[i]}; \sigma_{\beta^*}(X)) \leq \sqrt{\epsilon}. \quad (9.47)$$

Showing that $\lim_{\Omega \rightarrow 0} \text{Corr}(X_{[i]}; \sigma_{\beta^*}(X)) = 0$. When $X_{[i]}$ is not constant, this also implies $\lim_{\Omega \rightarrow 0} \beta_{[i]}^* = 0$ $\square$

One empirical example of the consequences of theorem 9.5.3 is shown in fig. 9.2 where a Gaussian and a mixture of Gaussians are used to model $\mathbb{P}_{Z|S}$, and $\mathbb{E}_{Z|S}[Z]$ gradually converge to the same value, while the variance is preserved.

Similarly to section 9.4 we adopt a Gaussian target distribution for $\mathbb{P}_{Z|S}$. This is a particularly compelling choice from the lens of the maximum entropy principle.

The fairness penalty term can be obtained from the Kullback-Leibler (KL) divergence between a standard Gaussian $\mathcal{N}(0, 1)$ and $\mathcal{N}(\mu, \sigma^2)$, which is given by

$$\text{KL}(\mu, \sigma) = \frac{1}{2} (\sigma^2 + \mu^2 - 1 - \log \sigma^2).$$

Accordingly, the total penalty term can be obtained as

$$\mathcal{L}^{\text{SS}} = \sum_{s \in \mathcal{S}} \|\text{Var}[Z | S = s] + \mathbb{E}[Z | S = s] - 1 - \log(\text{Var}[Z | S = s])\|. \quad (9.48)$$

where $\text{Var}[Z | S]$ and $\mathbb{E}[Z | S]$ denote the empirical moments.

9.5.3 Fairness Guarantees

This approach marks a fundamental departure from popular methods in the literature, which primarily seek to minimize $\Delta(\mathbb{P}_{Z|S=0}, \mathbb{P}_{Z|S=1})$. Instead of eliminating all traces of the sensitive attribute S , our method allows S to remain encoded in the representation but in a form that is provably inaccessible to the classifier being trained.

By exploiting the convexity of LR, we can explicitly quantify the ϵ -fairness of a representation z . This establishes that no logistic classifier $\sigma_{\beta}(x)$ can extract more than an ϵ amount of information about the sensitive attribute s . Such a result provides substantially stronger guarantees than adversarial evaluation, which lacks both provable tightness and robustness.

When constrained by fixed first and second moments, the Gaussian distribution uniquely maximizes entropy.

9.6 Connections to Maximum Mean Discrepancy

The Maximum Mean Discrepancy (MMD) is a widely used measure of distance between probability distributions. Given a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$, it is defined as

$$\text{MMD}_{\mathcal{H}}(\mathbb{P}_X, \mathbb{P}_Y) = \sup_{\|f\|_{\mathcal{H}} \leq 1} |\mathbb{E}[f(X)] - \mathbb{E}[f(Y)]|, \quad (9.49)$$

For random variables $X, Y \in \mathbb{R}^d$, the MMD can be expressed in terms of characteristic functions (CFs) [CRSG15]:

$$\text{MMD}_{\mathcal{H}^2}(\mathbb{P}_X, \mathbb{P}_Y) = \int_{\mathbb{R}^d} |\varphi_X(t) - \varphi_Y(t)|^2 w(t) dt, \quad (9.50)$$

where $w(t)$ is the inverse Fourier transform of the kernel of $\mathcal{H}$. When w is a probability density function, the integral can be interpreted as an expectation, yielding

$$\text{MMD}_{\mathcal{H}}(\mathbb{P}_X, \mathbb{P}_Y) = \text{CFD}_{\mathbb{P}_T}(\mathbb{P}_X, \mathbb{P}_Y). \quad (9.51)$$

The formulation presented in section 9.5 shares similarities with the MMD penalty used by VFAE. Specifically, MMD asserts that two distributions are equal if the maximum discrepancy between their expectations under all transformations $f \in \mathcal{H}$ is zero. Thus, if $\text{MMD}_{\mathcal{H}}(\mathbb{P}_X, \mathbb{P}_Y) = 0$, then $\mathbb{P}_X = \mathbb{P}_Y$.

This principle is also exploited by adversarial learning, which attempts to find such a transformation f through min-max optimization. However, as shown by Balunovi et al. [BRV22], adversarial methods often fail because the two distributions change dynamically with the current choice of f , making convergence difficult. As a result, when $\text{MMD}_{\mathcal{H}}(\mathbb{P}_X, \mathbb{P}_Y) \neq 0$, no provable fairness guarantees can be established.

By contrast, when f is restricted to the family of LR, theorem 9.5.3 shows that the second moment of the representation bounds its predictive power with respect to the sensitive attribute. This condition is strictly weaker than distributional matching as in [BRV22], adversarial learning, or CVAE. Indeed, $\mathbb{P}_{Z|S=0} = \mathbb{P}_{Z|S=1} \Rightarrow \mathbb{E}[Z | S = 1] = \mathbb{E}[Z | S = 0]$ but not the other way around.

Yet, given LR has a convex loss function, the optimal adversarial classifier can be provably trained to convergence with a second-order optimizer, thus giving guarantees on the amount of sensitive S that can at most be used in the classification.

9.7 Matching All Moments

The approaches introduced in sections 9.4 and 9.5 can be interpreted as matching either the moment generating function or, equivalently, the moments of a distribution. In theory, matching all moments of two distributions with bounded support ensures equality almost everywhere. While it is trivial to enforce bounded support for Z , directly applying moment matching to $\mathbb{P}_{Z|S}$ is impractical: characterizing a distribution requires all moments, which is computationally infeasible. Moreover, truncating at the k -th moment introduces bias and, in this context, risks leakage of the sensitive attribute S .

A well-known tool for unbiased estimation of an infinite series of moments is the *Russian Roulette estimator*. For an infinite series $Y = \sum_{i=1}^{\infty} Y_i$, the estimator randomly selects a truncation point N and computes the partial sum up to N , reweighting each term to preserve unbiasedness:

$$\hat{Y} = \sum_{i=1}^N \frac{Y_i}{P(N \geq i)}, \quad (9.52)$$

Although unbiased, the estimator can suffer from high variance if truncation probabilities are not carefully chosen, making it challenging to balance computational efficiency with stability in practice.

In contrast, the proposed methods in sections 9.4 and 9.5 offer a more principled and computationally stable strategy for matching distributions, avoiding the pitfalls of moment truncation and high-variance estimators.

9.8 Experimental Evaluation

FNF [BRV22] offers the strongest theoretical and empirical guarantees, making it the current state of the art in fair representation learning. Therefore, we primarily evaluate the performance of our proposed methods on the same suite of benchmarks, while also considering additional methods and datasets for a more comprehensive analysis. We utilize six well-known datasets in our study, sourced from both the UCI Machine Learning Repository and other publicly available resources. These datasets include Adult, Crime, Compas, Law School, Health, and the Statlog (German Credit Data) dataset. Each of these datasets presents distinct challenges and characteristics relevant to fairness and predictive modeling. Below, we briefly introduce each dataset and outline the preprocessing steps employed:

- Adult (archive.ics.uci.edu/adult): The Adult dataset, also known as the Census Income dataset, originates from the 1994 Census database and is available through the UCI repository [DG17b]. It contains 14 attributes, including age, workclass, education, marital status, occupation, race, and sex. The prediction task is to determine whether an individual's income exceeds \$50,000 per year. To facilitate modeling, we discretize continuous features and retain categorical variables related to demographics and employment. Sex is treated as the protected attribute in fairness analyses.
- Crime (archive.ics.uci.edu/communities+and+crime): The Communities and Crime dataset combines socio-economic data from the 1990 US Census, law enforcement data from the 1990 LEMAS survey, and crime data from the 1995 FBI UCR. The goal is to predict whether the violent crime rate of a community is above or below the median. We utilize attributes such as race percentages, income levels, and family structure indicators. Race is designated as the protected attribute, derived from the proportions of racial groups within each community.
- Compas (github.com/propublica/compas-analysis): The Compas dataset contains data related to criminal history, jail and prison time, demographics, and COMPAS risk scores from Broward County (2012-2013) [BDE09]. The prediction task involves forecasting recidivism within two years. Key attributes include age, prior count, and charge degree, with race being the protected attribute. We preprocess the dataset by discretizing continuous variables and retaining categorical ones, critical for risk prediction.
- Health (paperswithcode.com/heritage-health-prize): The Health dataset was part of the Heritage Health Prize competition on Kaggle and contains medical records of over 55,000 patients. We focus on the merged claims, drug count, and lab count attributes while removing personal identifiers to ensure privacy. Age is treated as a protected attribute, divided into binary groups: above and below 60 years. The primary prediction task is to assess the maximum Charlson Comorbidity Index, reflecting the long-term survival prospects of patients with multiple conditions.
- Law School: The Law School dataset consists of data from admissions

cycles between 2005 and 2007, covering over 100,000 individual applications. Attributes include LSAT scores, undergraduate GPA, race, gender, and residency status. To enhance privacy, data has been aggregated where necessary. We consider race as the protected attribute, binarizing it into white and non-white categories. The main task is to predict law school admission outcomes.

- German (archive.ics.uci.edu/south+german+credit): The German Credit Data dataset, sourced from the UCI Machine Learning Repository [DG17b], consists of 1,000 records of credit applicants. Each instance is labeled as either "good" or "bad" credit risk. The dataset contains 20 features, including age, credit amount, employment status, and personal status, among others. Both categorical and numerical data are present, requiring careful preprocessing. We encode categorical variables using one-hot encoding and normalize numerical features. The protected attribute for fairness evaluation in this dataset is age, segmented into groups representing different age ranges.

In [BRV22], the Lawschool dataset was also used to demonstrate the effectiveness of FNF. Despite extensive investigation, we did not find any publicly available version of this dataset, and therefore our approaches were not evaluated on it.

Across all tested datasets, both of our proposed methods match or surpass state-of-the-art approaches in terms of accuracy while producing fairer representations. Interestingly, even though the method presented in section 9.5 only provides guarantees against the worst-case linear attacker, it also performs strongly under adversarial evaluations with deep neural networks. Additional experimental results are reported, including further Pareto plots obtained under different hyperparameter settings for both FmSS (section 9.5) and FmCF (section 9.4).

All reported results are averages over 10 different seeds of the proposed approaches. No extensive hyperparameter tuning was performed for any of the reported results, underscoring the effectiveness of the proposed methods. All MLPs used for adversarial evaluations, encoding, and classification consist of four layers with 64 neurons each. Additionally, larger MLPs for fairness adversarial evaluations, configured with 128 and 256 neurons, were tested but resulted in poorer performance. The Adam optimizer [KB14] was employed for all training sessions, with a learning rate of 0.0003. Training was conducted for 100 epochs, incorporating L2 regularization with a weight

penalty of 0.0001 to mitigate overfitting. Since accuracy and fairness often involve a trade-off, early stopping was not applied. Instead, the model from the final epoch was used for evaluation.

9.8.1 Visualization of Learned Representations

In order to have a better sense of what the CF approach presented in section 9.4 is learning, we visualize the learned representation in fig. 9.3.

Indeed, it can be seen that for both datasets, the distribution $\mathbb{P}_{Z|S}$, which is the distribution of interest to evaluate the fairness of a representation, is very close to being a Gaussian. Thus, the CF is effectively enforcing the similarity to such a distribution. Instead, $\mathbb{P}_{Z|Y}$, which instead is the distribution of interest learn the downstream task, does not resemble a Gaussian distribution, and in particular, while $\mathbb{P}_{Z|S=0} \approx \mathbb{P}_{Z|S=1}$, $\mathbb{P}_{Z|Y=0} \not\approx \mathbb{P}_{Z|Y=1}$.

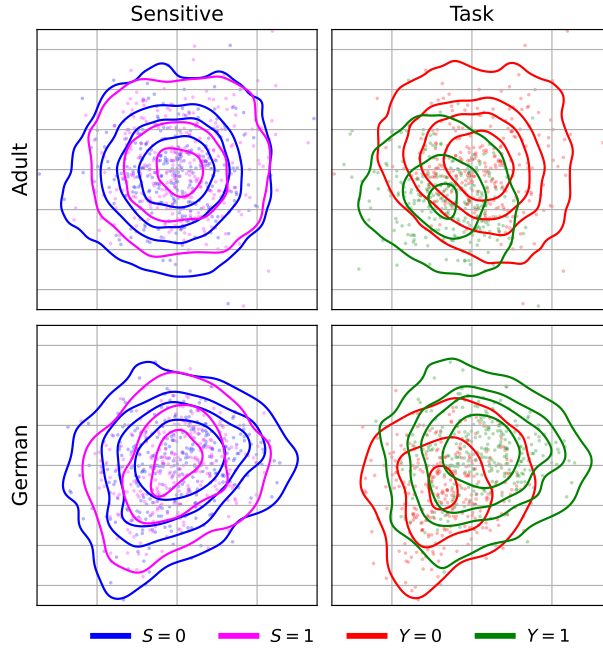


Figure 9.3: Latent distributions of $\mathbb{P}_{Z|S}$ and $\mathbb{P}_{Z|Y}$ obtained using FmCF.

9.8.2 Comparison on Adult Dataset

The Adult Dataset dataset is the most fundamental benchmark in fairness, as almost all methods in literature are evaluated on it [JMVS⁺20, RB19,

Table 9.1: Comparison on Adult dataset.

Model	Acc.	Adv. Accuracy	
		$g \in \mathcal{G}$	$g \notin \mathcal{G}$
Random	75.20	50.00	50.00
AdvForgetting [JMVS ⁺ 20]	85.99	66.68	74.50
MaxEnt-ARL [RB19]	84.80	69.47	85.18
LAFTR [MCPZ18]	86.09	72.05	84.58
FNF [BRV22]	84.43	N/A	59.56
FmCF (ours)	85.01	54.92	56.64
FmSS (ours)	85.02	51.70	57.47

Table 9.2: Comparison on Adult dataset using the setup from [MGB⁺18].

Model	Acc.	Adv. Acc.
Random	75.2	67.5
VFAE [LSL ⁺ 17]	84.2	88.2
CIAFL [XDD ⁺ 17]	83.1	88.8
IRwAL [MGB ⁺ 18]	84.2	77.6
FmCF (ours)	85.0	67.8
FmSS (ours)	85.0	67.9

MCPZ18, BRV22]. It requires predicting the income of a person without considering their gender [DHMS21]. For this dataset, we use the same setup as [BRV22], by evaluating our approach both on a classifier in the same family of the classifier used for the task, and one from a family of much more expressive ones, thus deeper and wider. This allows for a much tighter evaluation of the statistical distance $\Delta(\mathbb{P}_{Z|S=0}, \mathbb{P}_{Z|S=1})$. However, in we also compare to other baselines that only offer guarantees on the trained family of functions. For the proposed methods in table 9.1 $g \in \mathcal{G}$ is the class of LR classifiers, while $g \notin \mathcal{G}$ are deep MLPs. By construction, FmCF should be evaluated for $g \notin \mathcal{G}$, while FmSS for $g \in \mathcal{G}$.

In table 9.2 we compare our proposed approaches to other baselines on the Adult dataset, with the same setup as in [MGB⁺18]. Given that no distinction is made on the family of functions used for the evaluation, we report the Adv. Accuracy of FmCF and FmSS approaches using a deep and wide MLP.

9.8.3 Comparison on German Dataset

Table 9.3: Comparison on German dataset.

Model	Acc.	Adv. Acc.
Random	69.00	70.00
VFAE [LSL ⁺ 17]	72.00	71.70
CIAFL [LSL ⁺ 17]	69.50	81.10
IRwAL [MGB ⁺ 18]	71.00	69.80
FmCF (ours)	74.60	69.00
FmSS (ours)	74.20	69.00

The German dataset has been explored in the literature under different settings. In particular, [MGB⁺18] considers gender as the sensitive attribute, with the task of predicting income based on a set of personal features. To ensure a fair comparison, we evaluate our approach against several existing methods using the same experimental setup as in [MGB⁺18], as presented in table 9.3. The proposed method outperforms the other approaches both in terms of accuracy and fairness, showcasing the effectiveness of the novel penalties.

In table 9.4 we compare our proposed approaches to other baselines on the German dataset, with the same setup as in [TS22]. Given that no distinction is made on the family of functions used for the evaluation, we report the Adv. Accuracy of FmCF and FmSS approaches using a deep and wide MLP.

Table 9.4: Comparison on German dataset using the setup from [TS22].

Model	Acc.	Adv. Acc.
Random	70.0	81.0
CSN [TS22]	73.1	81.3
CIAFL [XDD ⁺ 17]	73.6	81.1
VFAE [LSL ⁺ 17]	72.8	81.2
FRTrain [RLWS20]	72.7	80.9
WassDB [JPS ⁺ 20]	72.8	81.1
FmCF (ours)	74.1	81.1
FmSS (ours)	74.2	81.1

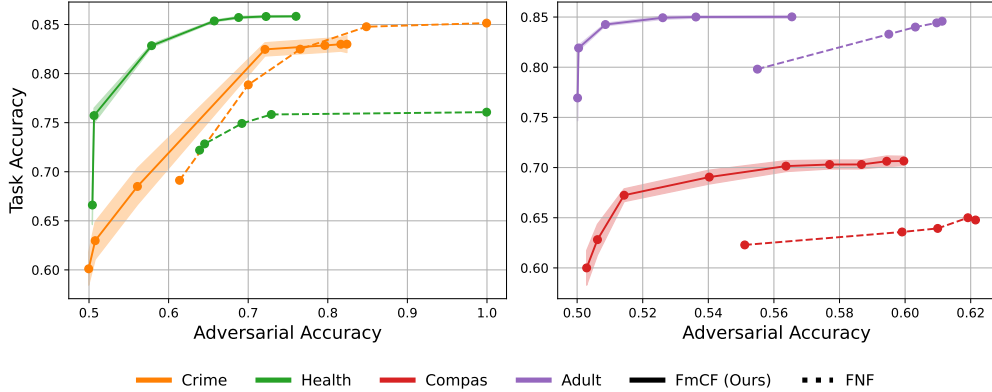


Figure 9.4: Pareto front between FmCF and FNF comparing task accuracy and fairness (95% confidence intervals from 10 seeds).

9.8.4 Comparison with FNF

Since [BRV22] achieves the best balance between fairness and accuracy, in section 9.8.4 we compare our proposed method from section 9.4 against all datasets used in their evaluation. These include four distinct datasets: Crime and Health, which contain continuous features that NFs can handle directly, and Compas and Adult, which include discrete features requiring separate handling by Normalizing Flows.

The plots show that our proposed methods consistently match or outperform [BRV22], while offering the additional advantage of not requiring access to the sensitive attribute during evaluation. Further results report Pareto fronts obtained by varying the latent dimension size, illustrating its effect on performance.

In figs. 9.5a to 9.5d, we present the trade-offs between adversarial balanced accuracy and task accuracy. For FmCF (section 9.4), we evaluate fairness against an adversarial MLP. For FmSS (section 9.5), we evaluate against both an MLP and a LR. In all figures, Fair Normalizing Flows [BRV22] are shown in black, while our method is shown in blue (1-dimensional latent space), orange (2 dimensions), and green (3 dimensions).

A clear trend emerges: the lower the latent dimension, the more effective the penalties become, yielding surprisingly strong results even for extremely low-dimensional representations. Importantly, this reduction in dimensionality does not limit downstream task performance. Across all tested configurations, both of our approaches still surpass the baselines.

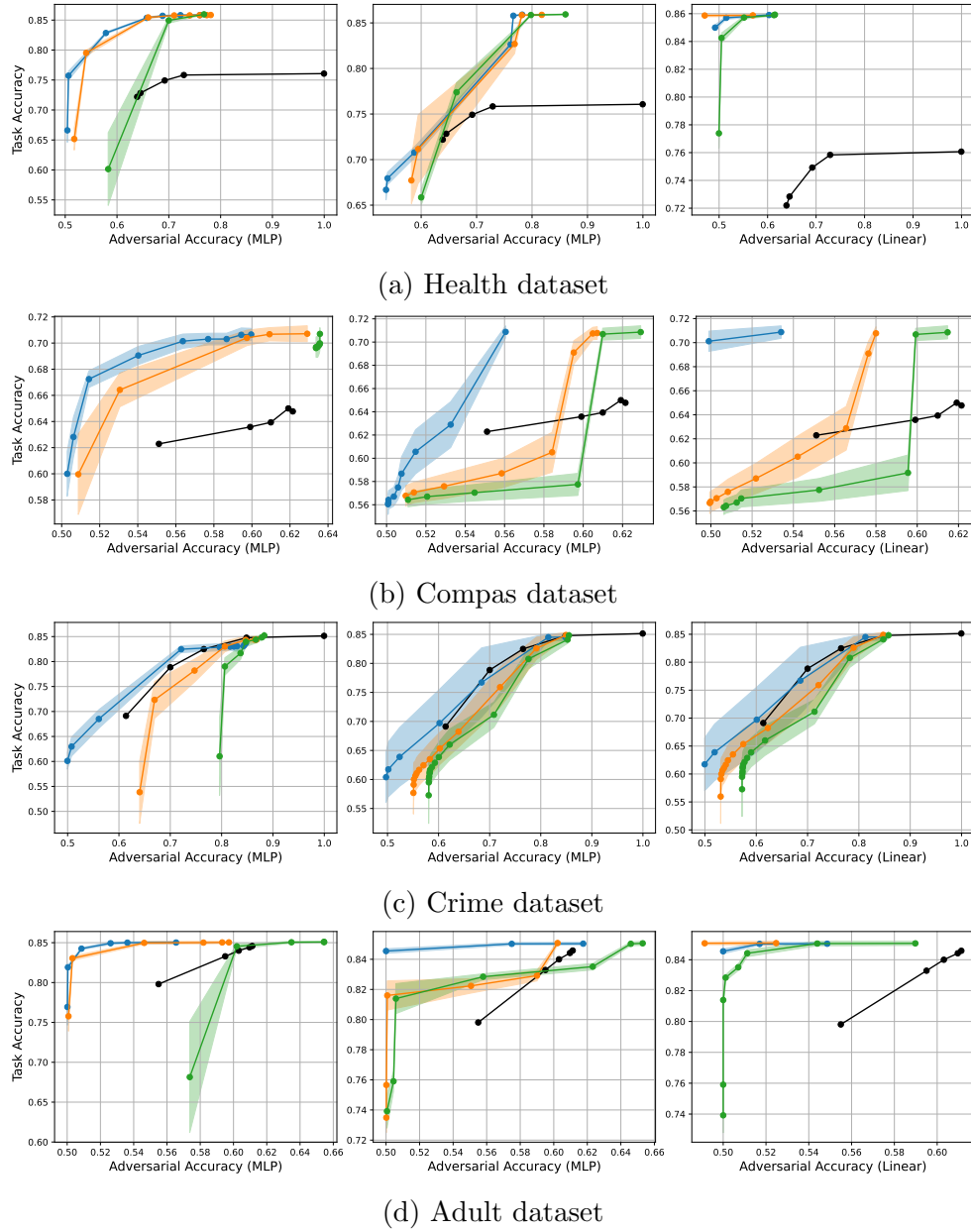


Figure 9.5: Pareto varying latent dimension across different datasets.

9.8.5 Evaluation of Fair Generative Models

Thus far, all benchmarks have focused on classification tasks. To illustrate the broader applicability of FmCF, we present a simple example in fig. 9.6. Here, a conditional convolutional autoencoder is trained on the MNIST handwritten digits dataset to learn a latent representation. The loss function described in algorithm 1 is then applied to penalize the divergence between $\mathbb{P}_{Z|S}$ and a Gaussian distribution. Enforcing this condition ensures that $Z \perp S$, i.e., the latent representation Z captures information from X that is independent of S .

As a result, when the model is tasked with reconstructing an input under a different label, it preserves the essential characteristics of the original input while generating a digit consistent with the new label. In practice, the decoder successfully retains properties such as rotation while relying solely on the provided label to determine the digit identity.

9.9 Conclusions

In this work, we introduced a novel framework for learning fair specialized representations using CFD, addressing key limitations of adversarial and normalizing flow-based approaches. By leveraging characteristic functions, our method achieves stable and efficient fairness by minimizing sensitive information leakage while preserving high predictive performance. We further proposed a simplified variant of the framework that provides certifiable bounds on the sensitive information accessible to downstream tasks. Experimental results demonstrate that our approaches consistently match or outperform state-of-the-art methods in the fairness-accuracy trade-off across multiple benchmark datasets. Importantly, unlike many existing approaches, our method does not require access to sensitive attributes at inference time, making it well suited for real-world applications.

Beyond empirical performance, our approach offers a principled, non-adversarial alternative to existing paradigms, opening a new avenue for research in fair representation learning. Nonetheless, like many fairness-focused algorithms, it faces challenges in high-dimensional settings, where the effectiveness of CFD may degrade. This limitation highlights the need for future work on improving the scalability and robustness of the framework. Moreover, the field as a whole is constrained by fairness benchmarks that primarily focus on low-dimensional datasets, limiting comprehensive evaluation under more realistic conditions. Developing richer, high-dimensional fairness



Figure 9.6: Latent representations Z are extracted from the left-most column and combined with arbitrary target labels to generate stylistically consistent images of different classes, shown on the right.

benchmarks would substantially advance the assessment of methods such as ours and foster further progress in the field.

10

Monotonic Neural Networks

10.1 Introduction

Monotonicity is a property requirement that often arises in many practical scenarios. This is especially the case in high stakes scenarios such as safety-critical systems, where increasing a control parameter (i.e. braking force) needs to have a predictable effect on the output of the system. This principle can also be applied to predictive models, for instance, as the age of a component or the number of operation cycles increases, the failure probability or remaining useful life estimate should also increase. Beyond pure control system setting, monotonicity is useful also in medical applications, where an increase in biomarkers should relate monotonically with predicted risk of a disease or with the intensity of the recommended intervention. Finally, credit and insurance are also critical applications where monotonicity is often required. A borrower with a higher income or lower debt-to-income ratio should never be assigned a higher risk score than someone with clearly worse financial indicators. Similarly for insurance pricing, higher risk factors should not lead to lower premiums.

By guaranteeing that outputs evolve consistently with respect to key inputs, monotonic models offer critical reliability in these critical tasks. In addition to robustness and reliability, these structural guarantees enhance

interpretability, support auditing and verification, and reduce the likelihood of harmful spurious patterns, strengthening fairness.

Monotonic neural networks represent a pivotal shift in deep learning, bridging the gap between high-capacity non-linear models and the need for interpretable, consistent outputs in practical applications. Monotonic MLPs preserve monotonic input-output relationships, making them particularly suitable for domains that demand justified and transparent decisions [GCP⁺16, NM19]. Trust and fairness are rapidly becoming critical in Artificial Intelligence technologies, and enforcing structural constraints on models is central to guaranteeing such desirable properties.

Monotonic MLPs have also been leveraged to design novel architectures for density estimation [CS20, OA⁺19, TLP19], survival analysis [JYTB23], and remaining useful life prediction [SCC23]. More broadly, the enforcement of architectural constraints ensures fairness, robustness, or other properties of interest. Explicitly embedding inductive biases that exploit prior knowledge has also proven fundamental for efficient generalization [DBB⁺00, MFCC⁺16, YDC⁺17]. Consequently, monotonic networks can improve both performance [Mit80] and data efficiency [Vel19].

Recent contributions to this field generally fall into two categories: soft monotonicity and hard monotonicity. Soft monotonicity relies on optimization constraints [GSM⁺19, SAM96], typically in the form of additional penalty terms in the loss. These approaches are simple to implement, computationally inexpensive, and benefit from the expressive power of Multi-Layer Perceptrons (MLPs) to approximate arbitrary functions. However, since penalties are applied to dataset samples, monotonicity is enforced only in-distribution and struggles to generalize out-of-distribution.

Hard monotonicity, in contrast, embeds monotonicity directly into the architecture [WL19, NKW23]. The simplest approach constrains MLP weights to be non-negative and employs monotonic activation functions [DV10]. Methods exploiting this parameterization [DV10, WL19] generally require bounded activations such as sigmoid or hyperbolic tangent, which are known to pose optimization challenges due to vanishing gradients [DSC22, RS23a, Sza21, NH10, GB10, GWFM⁺13]. This issue is even more pronounced in monotonic networks with non-negative weights, where initialization becomes critical for effective optimization. As discussed in section 10.11, poor initialization may cause saturated activations early in training, significantly slowing convergence. Moreover, the reliance on bounded activations restricts MLPs to representing only bounded functions, potentially limiting generalization, as

illustrated in fig. 10.1.

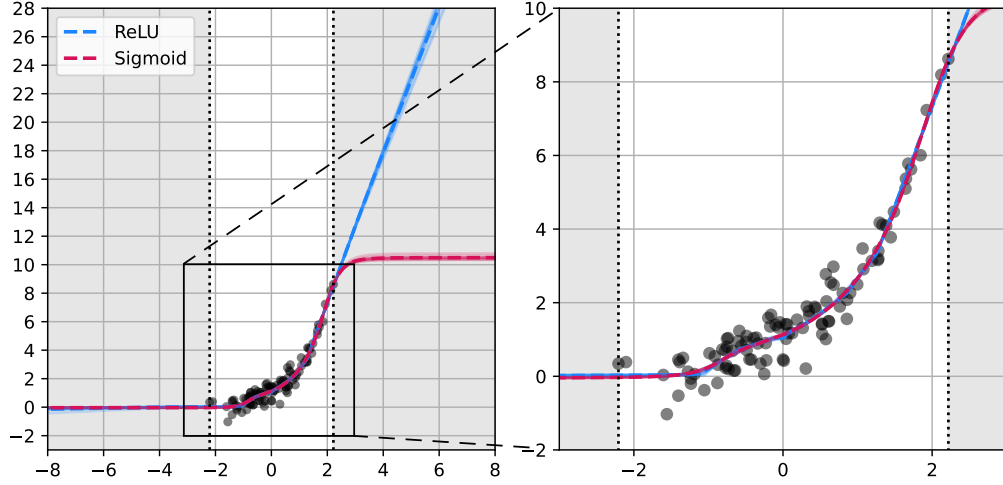


Figure 10.1: Monotone MLPs with weight-constraint and bounded activations (pink) and our proposed approach based on ReLU (blue). The former can only represent bounded functions and, thus, cannot extrapolate the data trend, which is important in many domains, such as time-series analysis and predictive maintenance.

Indeed, most recent advances in neural networks employ activations from the family of rectified linear functions, such as the widely used ReLU activation [Vas17, HZRS16]. However, using ReLU activations in MLPs with non-negative weights is problematic. Specifically, an MLP that combines convex activations (such as ReLU) with non-negative weights can only approximate convex functions, which severely limits its applicability [DV10, MR22]. For this reason, many approaches in the literature continue to rely on bounded activations to ensure the universal approximation capabilities of the network.

Conventional techniques for enforcing monotonicity in MLPs by design typically employ non-negative weight constraints and bounded activations, both of which introduce well-known optimization challenges. In this work, we generalize previous theoretical results, showing that MLPs with non-negative

weight constraints and activations that saturate on alternating sides are universal approximators of monotonic functions. Additionally, we establish an equivalence between the saturation side of the activations and the sign of the weight constraint. This connection allows us to prove that MLPs with convex monotone activations and non-positive weight constraints are also universal approximators, in contrast to their non-negative constrained counterparts. These results provide theoretical grounding for the empirical effectiveness observed in prior work, while also suggesting possible architectural simplifications. To further mitigate optimization difficulties, we propose an alternative formulation that allows the network to adapt its activations according to the sign of the weights. This approach removes the need for weight reparameterization, easing initialization and improving training stability. Experimental evaluations confirm the validity of our theoretical findings, demonstrating that our approach compares favorably with traditional architectures.

The primary goal of this work is to extend the theoretical foundation of monotonic-constrained MLPs by showing that universal approximation can still be achieved with activations that saturate on one side, such as ReLU. To ensure that these findings have practical significance, we introduce a new architecture that relies exclusively on saturating activations, achieving performance comparable to state-of-the-art methods.

Our contributions can be summarized as follows:

1. We show that constrained MLPs that alternate left-saturating and right-saturating monotonic activations can approximate any monotonic function. We also demonstrate that this can be achieved with a constant number of layers, which matches the best-known bound for threshold-activated networks.
2. Contrary to the non-negative-constrained formulation, we prove that an MLP with at least 4 layers, non-positive-constrained weights, and ReLU activation is a universal approximator. More generally, this holds true for any saturating monotonic activation.
3. We propose a simple parametrization scheme for monotonic MLPs that (i) can be used with saturating activations; (ii) does not require constrained parameters, thus making the optimization more stable and less sensitive to initialization; (iii) does not require multiple activations; (iv) does not require any prior choice of alternation of any activation and its point reflection.

Our discussion will primarily focus on ReLU activations, which are among the most widely used in modern deep learning. Nevertheless, our results extend to the broader family of monotonic activations that saturate on at least one side. This family includes many ReLU-like variants, such as exponential, ELU [CUH16], SeLU [KUMH17], CELU [Bar17], SReLU [JXF⁺16], and others.

10.2 Related work

Monotonicity in neural network architectures is an active area of research, studied from both theoretical and practical perspectives. Prior work can be broadly categorized into two approaches: architectures designed with built-in constraints (hard monotonicity) and those that rely on regularization or heuristic techniques to enforce monotonicity (soft monotonicity). Our contribution falls into the former category.

10.2.1 Hard Monotonicity

Hard-monotonicity focuses on designing MLPs that are provably monotonic over the entire input space by construction, ensuring that only monotonic functions can be learned. Early examples include Deep Lattice Networks [YDC⁺17] and methods that constrain all weights to have the same sign [DBB⁺09, RS23b, KL24]. However, enforcing non-negative parameters violates the original MLP formulation, thereby invalidating the universal approximation theorem. In fact, universal approximation under this constraint has only been proven when the threshold function is used as activation and the network has at least four layers [RS23b]. Moreover, non-negative weight constraints create additional challenges for initialization, since common initializers may not be well-suited for positive semidefinite matrices.

More recent work has proposed new architectures to address these limitations, including adaptations of the Deep Lattice framework to MLPs [YMK22], approaches that employ multiple activations [RS23b], and methods that constrain the Lipschitz constant [RPK⁺17]. However, each of these introduces its own limitations: [RS23b] requires multiple activations and a pre-defined neuron split across layers, which may be suboptimal or demand additional tuning, while [NKG23] relies on very specific activations to achieve control, as noted by the authors. In contrast, our work seeks to overcome these drawbacks by improving flexibility while preserving strict monotonicity guarantees.

10.2.2 Soft Monotonicity

Soft-monotonicity approaches enforce monotonicity through modifications to the training process rather than the architecture itself, typically via heuristics or regularization. Examples include point-wise penalties for negative gradients [GSM⁺19, SAM96] and certification methods based on Mixed Integer Linear Programming (MILP) [LHZL20]. While these methods preserve high expressive power, they cannot guarantee monotonicity. Furthermore, certification procedures, such as those based on MILP or Satisfiability Modulo Theories (SMT) solvers, often incur prohibitively high computational costs.

10.3 Monotone MLP

A function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is said to be monotone non-decreasing with respect to x_i , if given $x_i^0, x_i^1 \in \mathbb{R}, i \in [1, d]$, has the following property:

$$x_i^0 \leq x_i^1 \Rightarrow f(x_1, \dots, x_i^0, \dots, x_d) \leq f(x_1, \dots, x_i^1, \dots, x_d) \quad (10.1)$$

And similarly, a function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is said to be monotone non-increasing with respect to x_i if:

$$x_i^0 \leq x_i^1 \Rightarrow f(x_1, \dots, x_i^0, \dots, x_d) \geq f(x_1, \dots, x_i^1, \dots, x_d). \quad (10.2)$$

Remark 10.3.1. Given $f(x), g(x)$ monotonic non-decreasing, and $h(x), u(x)$ monotonic non-increasing, $f \circ g$ is monotonic non-decreasing, $f \circ h$ is monotonic non-increasing, and $u \circ h$ is monotonic non-decreasing.

In this work, we will only focus on parametrizing non-decreasing functions, as the monotonicity can be reversed by simply inverting the sign of the inputs.

An MLP is defined as a parametrized function f_θ obtained as composition of alternating affine transformations $l_\theta^{(i)}$ and non-linear activations $\sigma_\theta^{(i)}$:

$$f_\theta(x) = l_\theta^{(1)} \circ \sigma_\theta^{(1)} \dots \sigma_\theta^{(n-1)} \circ l_\theta^{(n)}. \quad (10.3)$$

A straightforward way to construct a provably monotonic MLP that satisfies eq. (10.1) is to impose constraints on both weights and activations, ensuring that each component of eq. (10.3) is monotonic. For affine transformations $l_\theta^{(i)}(x) = W^{(i)}x + b^{(i)}$, it suffices to enforce a non-negative Jacobian,

In the rest of the work we will use “monotonic” as shorthand for “monotonic non-decreasing”, unless otherwise specified

which in this case reduces to requiring the weight matrix $W^{(i)}$ to be non-negative. Monotonic activation functions are, by definition, already monotonic.

To optimize such an MLP using standard unconstrained gradient-based methods, the non-negative weight constraint is typically enforced via reparameterization, i.e., $l_{\theta}^{(i)}(x) = g(W^{(i)}x + b^{(i)})$ for some differentiable $g : \mathbb{R} \rightarrow \mathbb{R}_+$. Common choices for g include the absolute value and squaring.

10.4 Known conditions for universal approximation

Despite their strong performance, a critical limitation of existing MLP architectures based on weight constraints is the restricted choice of activation functions. Constrained MLPs are known to be universal function approximators for monotonic functions, provided that the activation is the threshold function and the number of hidden layers exceeds the input dimension [DV10]. More recently, this requirement has been significantly improved to a constant bound, showing that four layers suffice to guarantee universal approximation when using the threshold activation [MR22]. As a result, practical implementations often resort to bounded activation such as sigmoid, tanh, or the ReLU6 function.

In contrast, the use of convex activations like ReLU in constrained MLPs drastically reduces network expressivity. This limitation can be seen from the following proposition:

Proposition 10.4.1. *The composition of monotonic convex functions is itself monotonic convex.*

Since affine transformations are both convex and concave, if the activation is chosen to be a monotonic convex function, then the constrained MLP can only approximate monotonic convex functions.

Despite this disadvantage, ReLU-activated monotonic MLPs remain of interest due to their favorable properties and strong empirical performance in unconstrained networks [GB10, HAB19]. In particular, [RS23b] propose a method to incorporate ReLU activations by combining multiple functions derived from a primitive activation $\sigma(x)$, its point reflection $\sigma'(x) = -\sigma(-x)$, and a bounded sigmoid-like activation. While ReLU activations are not central to the subsequent theoretical analysis, the bounded activation is essential to ensure universal approximation.

An upper bound on the minimum number of layers required for universal approximation in this architecture is inherited from [DV10]. Although this bound grows linearly with input dimensionality, the authors report good empirical performance with only a few layers. In this work, we extend the result of [MR22], proving that a constant number of layers is sufficient even when non-threshold activations are used. This result both explains the empirical findings of [RS23b] and suggests that one of the three activations employed in their architecture may be unnecessary.

10.5 Naive bound for universal approximation of alternating MLPs

A simpler, though looser, bound to prove that MLPs with alternating ReLU and its point reflection ReLU' activations are a universal monotonic function approximators can be achieved by building on the proof of [MR22]. Two simple observations are sufficient.

Remark 10.5.1. the composition of ReLU and its point reflection $\text{ReLU}'(x) = -\text{ReLU}(-x)$ can approximate the threshold function $\mathbb{1}_{x \geq 0}$ arbitrarily well:

$$\lim_{\alpha \rightarrow +\infty} \text{ReLU}(\text{ReLU}'(\alpha x) + 1) = \mathbb{1}_{x \geq 0}(x) \quad (10.4)$$

$$\lim_{\alpha \rightarrow +\infty} \text{ReLU}'(\text{ReLU}(\alpha x) - 1) = \mathbb{1}_{x \geq 0}(x) - 1 \quad (10.5)$$

A representation of eq. (10.4) is provided in fig. 10.2.

Remark 10.5.2. The formulas in eq. (10.4) can be implemented with a 2-layer constrained MLP, alternating ReLU and ReLU' activations.

This is enough to leverage the existing results for threshold-activated MLP [MR22]. This includes the best-known bound on the number of required hidden layers, which, however, doubles from 3 to 6 due to the need for two ReLU layers for the Heavyside approximation. However, this naive bound is unnecessarily loose, as shown in theorem 10.6.5.

10.6 Universal approximation theorem for non-threshold activations

From proposition 10.4.1, follows that a constrained MLP with ReLU activations cannot be a universal approximator for monotone functions. However, two MLP layers can approximate the Heavyside function arbitrarily well if

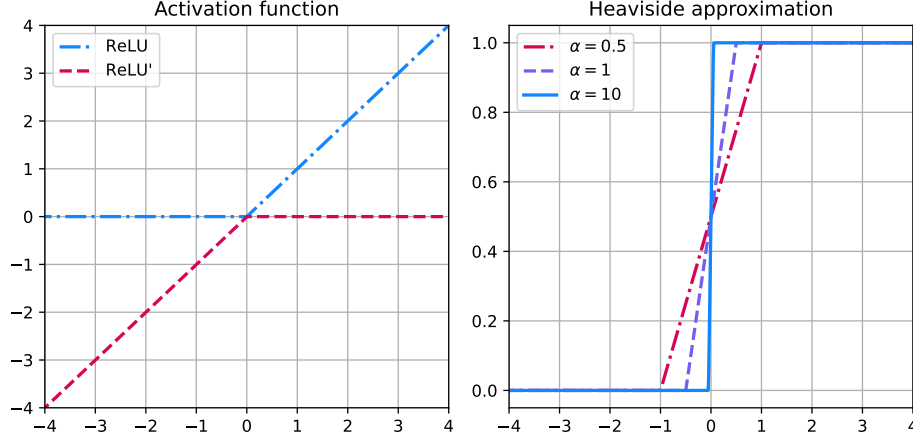


Figure 10.2: Constructions of Heaviside function using a composition of ReLU and its point reflection ReLU' to obtain $\text{ReLU}(\text{ReLU}'(\alpha x - 0.5) + 1) = \text{ReLU}'(\text{ReLU}(\alpha x + 0.5) - 1)$

the activations alternate between ReLU and its point reflection, as shown in fig. 10.2. From this observation, it is possible to use the result of [DV10] directly, showing that alternating activations between ReLU and its point reflection is sufficient to construct universal monotonic approximators. However, the resulting bound on the required number of layers is linear in the input dimension.

In section 10.5 we use similar reasoning and the results of [MR22] to achieve a loose but constant bound of 8 layers. However, in this section, we will derive a tighter bound that instead matches and generalizes the result derived in [MR22], while applying to a broader class of activation functions. Specifically, we will prove that 4 layers are sufficient, provided that the activations alternate saturation sides.

Definition 10.6.1. Given a function $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ saturates right/left if the corresponding limit exists and is finite. That is, σ is right-saturating if

$$\sigma(+\infty) := \lim_{x \rightarrow +\infty} \sigma(x) \in \mathbb{R},$$

and it is left-saturating if

$$\sigma(-\infty) := \lim_{x \rightarrow -\infty} \sigma(x) \in \mathbb{R}.$$

. We will denote the set of right-saturating activations as $\mathcal{S}^+$ and the set of left-saturating activations as $\mathcal{S}^-$.

Proposition 10.6.2. *For every MLP with non-negative weights and activation $\sigma(x)$, and for any $a \in \mathbb{R}_+, b \in \mathbb{R}$, there exists an equivalent MLP with non-negative weights and activation $a\sigma(x) + b$.*

In the following proofs, thanks to proposition 10.6.2, we will only consider activations that saturate to zero.

To prove the main result, we will first need to prove the following lemmas. The first step is proving that hidden units in the first layer can approximate piecewise-constant functions on specific half-spaces.

Lemma 10.6.3. *Consider an arbitrary hyperplane defined by $\alpha^T(x - \beta) = 0$, $\alpha \in \mathbb{R}_+^d$ and $\beta \in \mathbb{R}^d$, and the open half-spaces $A^+ = \{x : \alpha^T(x - \beta) > 0\}$, $A^- = \{x : \alpha^T(x - \beta) < 0\}$. The i -th neuron in the first hidden layer of an MLP with non-negative weights can approximate :*

$$h_i^{(1)}(x) \approx \begin{cases} \sigma^{(1)}(+\infty), & \text{if } x \in A^+ \\ \sigma^{(1)}(-\infty), & \text{if } x \in A^- \\ \sigma^{(1)}(0), & \text{otherwise} \end{cases}$$

Proof. Denote by w the weights and by b the bias associated with the hidden unit under consideration. Then, setting the parameters to $w = \lambda\alpha^T$ and $b = \lambda\alpha^T\beta$ and taking the limit we have that:

$$h_i^{(1)}(x) \approx \lim_{\lambda \rightarrow +\infty} \sigma^{(1)}(wx + b) = \lim_{\lambda \rightarrow +\infty} \sigma^{(1)}(\lambda\alpha^T(x - \beta))$$

The limit is either $\sigma^{(1)}(+\infty)$, $\sigma^{(1)}(-\infty)$ or $\sigma^{(1)}(0)$ depending on the sign of $\alpha^T(x - \beta)$, proving that:

$$h_i^{(1)}(x) \approx \begin{cases} \sigma^{(1)}(+\infty), & \text{if } \alpha^T(x - \beta) > 0 \\ \sigma^{(1)}(-\infty), & \text{if } \alpha^T(x - \beta) < 0 \\ \sigma^{(1)}(0), & \text{if } \alpha^T(x - \beta) = 0 \end{cases}$$

□

Note that $\sigma^{(1)}(\pm\infty)$ need not be finite.

The second step is to prove that one hidden layer can perform intersections of subspaces under specific conditions. In our construction, these will be either half-spaces or intersections of specific half-spaces.

Lemma 10.6.4. *Consider the intersection $A = \bigcap_{i=0}^n A_i$, for $A_1, \dots, A_n$ subsets of $\mathbb{R}^d$. For any γ in the image of $\sigma^{(k)}$, a single unit $h_j^{(k)}$ in the k -th hidden layer of an MLP with non-negative weights can approximate:*

$$h_j^{(k)}(x) \approx \gamma \mathbb{1}_A(x)$$

provided that $h_i^{(k-1)}(x) \approx 0$ for $x \in A_i$, and either:

- $\sigma^{(k)} \in \mathcal{S}^-$ and $h_i^{(k-1)}(x) < 0$ for $x \notin A_i$
- $\sigma^{(k)} \in \mathcal{S}^+$ and $h_i^{(k-1)}(x) > 0$ for $x \notin A_i$

Proof. Denote by w the weights and by b the bias associated with the hidden unit under consideration. Then, setting the weights to $w = \lambda \mathbf{1}^T$ and taking the limit we have that:

$$h_j^{(k)}(x) \approx \lim_{\lambda \rightarrow +\infty} \sigma^{(k)}(w h^{(k-1)}(x) + b) \quad (10.6)$$

$$= \lim_{\lambda \rightarrow +\infty} \sigma^{(k)}\left(b + \lambda \sum_{i=0}^n h_i^{(k-1)}(x)\right) \quad (10.7)$$

Note that in any case, if $x \in \bigcap_{i=1}^n A_i$, then $\lambda \sum_i h_i^{(k-1)}(x) \approx 0$, and the limit simply reduces to $\sigma^{(k)}(b)$. On the other hand, for $x \notin \bigcap_{i=1}^n A_i$, the limit can be either $\sigma^{(k)}(\pm\infty)$ depending on the sign of $h_i^{(k-1)}(x)$. When $\sigma^{(k)} \in \mathcal{S}^-$ and $h_i^{(k-1)}(x) < 0$, the limit is simply $\sigma^{(k)}(-\infty) = 0$. Similarly, when $\sigma^{(k)} \in \mathcal{S}^+$ and $h_i^{(k-1)}(x) > 0$ the limit is $\sigma^{(k)}(+\infty) = 0$.

In both cases, for any γ in the image of $\sigma^{(k)}$ we can find a bias value b so that:

$$h_j^{(k)}(x) \approx \gamma \mathbb{1}_A(x) = \begin{cases} \sigma^{(k)}(b) = \gamma, & \text{if } x \in \bigcap_{i=1}^n A_i, \\ \sigma^{(k)}(\pm\infty) = 0, & \text{otherwise} \end{cases}$$

□

Thanks to lemma 10.6.3 and lemma 10.6.4, we can now prove the main result, that is theorem 10.6.5. We will only prove the case $\sigma^{(1)} \in \mathcal{S}^-$, $\sigma^{(2)} \in \mathcal{S}^+$, $\sigma^{(3)} \in \mathcal{S}^-$. The proof for the opposite case follows the same structure.

We are now ready to prove the main result which is the following theorem.

Theorem 10.6.5. *An MLP $g_\theta : \mathbb{R}^d \rightarrow \mathbb{R}$ with non-negative weights and 3 hidden layers can interpolate any monotonic non-decreasing function $f(x)$ on any set of n points, provided that the activation functions are monotonic non-decreasing and alternate saturation sides. That is, in addition to monotonicity, either of the following holds:*

$$\sigma^{(1)} \in \mathcal{S}^-, \sigma^{(2)} \in \mathcal{S}^+, \sigma^{(3)} \in \mathcal{S}^- \quad (10.8)$$

$$\sigma^{(1)} \in \mathcal{S}^+, \sigma^{(2)} \in \mathcal{S}^-, \sigma^{(3)} \in \mathcal{S}^+ \quad (10.9)$$

Proof. Assume, without loss of generality, that the points $x_1, \dots, x_n$ are ordered so that $i < j \implies f(x_i) \leq f(x_j)$, with ties resolved arbitrarily. We will proceed by construction, layer by layer.

Layer 1 Since the function to interpolate is monotonic, for any couple of points x_i, x_j with $i < j$, it is possible to find a hyperplane defined by $\alpha_{j/i}^T (x - \beta_{j/i}) = 0$ for some $\beta_{j/i} \in \mathbb{R}^d$ and some non-negative normal $\alpha_{j/i} \in \mathbb{R}_+^d$, such that $x_j \in A_{j/i}^+, x_i \in A_{j/i}^-$, where $A_{j/i}^+$ and $A_{j/i}^-$ denote the positive and negative half spaces respectively.

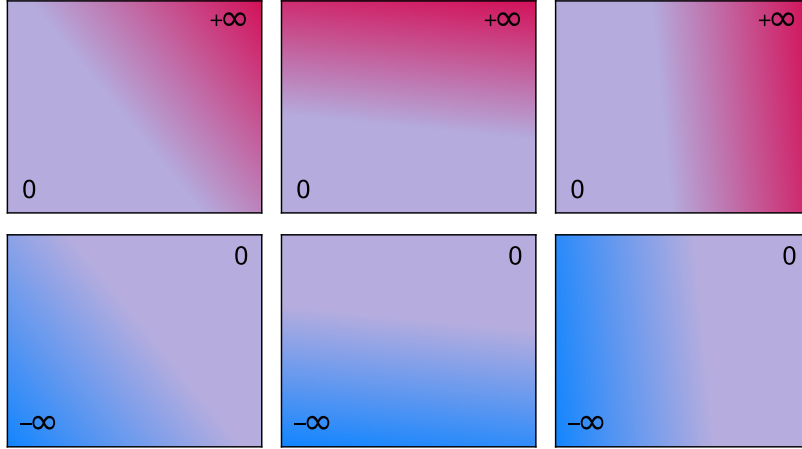


Figure 10.3: Example of representable functions at layer 1.

Using lemma 10.6.3, we can ensure that it is possible to have:

$$\begin{cases} h_i^{(1)}(x) \approx \sigma^{(1)}(-\infty) = 0, & \text{if } x \in A_{j/i}^- \forall i, j \\ h_i^{(1)}(x) \approx \sigma^{(1)}(+\infty) > 0, & \text{otherwise} \end{cases} \quad (10.10)$$

A visual example is shown in fig. 10.3.

Layer 2 Let us construct the set $A_i^{(2)} = \bigcap_{j:j>i} A_{j/i}^-$. Note that the sets $A_i^{(2)}$ always contain x_i and do not contain any x_j for $j > i$. Using eq. (10.10), we can apply lemma 10.6.4, which ensures that it is possible to have the following:

$$\begin{cases} h_i^{(2)}(x) \approx 0, & \text{if } x \notin A_i^{(2)} \\ h_i^{(2)}(x) \approx \gamma^{(2)} < 0, & \text{otherwise} \end{cases} \quad (10.11)$$

A visual example is shown in fig. 10.4.

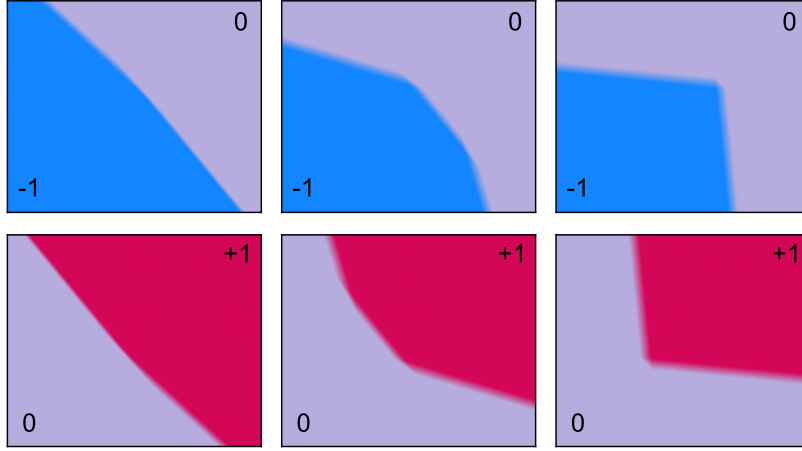


Figure 10.4: Example of representable functions at layer 2.

Layer 3 Consider $A_i^{(3)} = \bigcap_{j:j<i} \bar{A}_j^{(2)}$, where $\bar{A}_j^{(2)}$ is the complement of $A_j^{(2)}$. Using eq. (10.11) we can once again apply lemma 10.6.4, which ensures that it is possible to have the following:

$$h_i^{(3)}(x) \approx \gamma^{(3)} \mathbb{1}_{A_i^{(3)}}(x) \quad (10.12)$$

Now, we will show that $A_i^{(3)}$ represents a level set, i.e. $x_j \in A_i^{(3)} \iff f(x_j) \geq f(x_i)$. To do so, consider that $\bar{A}_i^{(3)} = \bigcup_{j:j<i} A_j^{(2)}$. Since $x_j \in A_j^{(2)}$, then $x_j \in \bar{A}_i^{(3)}$ for $j < i$. Similarly since x_j is the largest point contained in $A_j^{(2)}$, $\bar{A}_i^{(3)}$ cannot contain x_i or any point larger than x_i . This shows that $A_i^{(3)}$ contains exactly the points $\{x_j : f(x_j) \geq f(x_i)\}$. A visual example is shown in fig. 10.5.

In this case $\gamma^{(2)} < 0$ since by assumption $\sigma^{(2)}$ saturates right.

In this case $\gamma^{(3)} > 0$ since by assumption $\sigma^{(3)}$ saturates left.

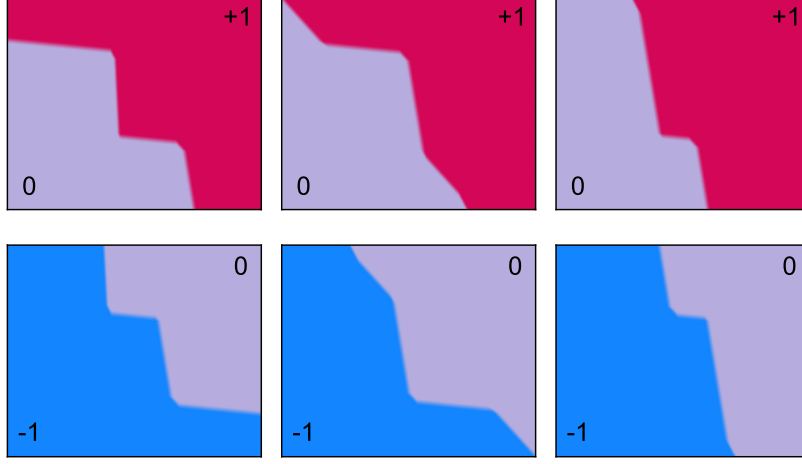


Figure 10.5: Example of representable functions at layer 3.

Layer 4 To conclude the proof, simply take the weights at the fourth layer to be :

$$w = \left[\frac{f(x_1) - b}{\gamma^{(3)}}, \frac{f(x_2) - f(x_1)}{\gamma^{(3)}}, \dots, \frac{f(x_n) - f(x_{n-1})}{\gamma^{(3)}} \right]$$

Since the points are ordered, this ensures that w contains all non-negative terms, when bias term b is taken to be $b \leq f(x_1)$. Defining $f(x_0) = b$, the output of the MLP can be expressed as:

$$g_\theta(x) = w^T h^{(3)}(x) + b = b + \sum_{j=1}^n (f(x_j) - f(x_{j-1})) \mathbb{1}_{A_j^{(3)}}(x) \quad (10.13)$$

Evaluating eq. (10.13) at any of the points x_i , it reduces to the telescopic sum:

$$g_\theta(x_i) = f(x_1) + \sum_{j=2}^i (f(x_j) - f(x_{j-1})) = f(x_i) \quad (10.14)$$

Thus proving that the network correctly interpolates the target function. $\square$

10.7 Non-positive constrained monotonic MLP

Consider the simple modification of the standard constrained MLP approach in eq. (10.3): instead of constraining the weights to be non-negative, they

are constrained to be non-positive. Although this change might appear inconsequential, we will show that it is not. In fact, a non-positive constrained MLP satisfies the conditions of theorem 10.6.5, provided the activation function saturates on at least one side. This includes convex activations such as ReLU, which are provably not universal approximators under non-negative weight constraints.

By theorem 10.3.1, an MLP defined as in eq. (10.3) remains monotone for an even number of non-positive layers. Thus, it is possible to construct provably monotonic networks using non-positive weight constraints. In what follows, we restrict attention to networks with an even number of layers, although the result also holds for odd depths.

A first key observation is that σ and its point reflection σ' share the same monotonicity but saturate in opposite directions.

Proposition 10.7.1. *If $\sigma(x)$ is monotonic non-decreasing, then its point reflection $\sigma'(x)$ is also monotonic non-decreasing. If $\sigma(x)$ saturates, then $\sigma'(x)$ also saturates, but in the opposite direction.*

From proposition 10.7.1, we obtain the following immediate corollary of theorem 10.6.5:

Proposition 10.7.2. *An MLP with at least 4 layers, non-negative weights, and alternating activations σ and σ' is a universal monotonic approximator, provided that σ saturates on at least one side.*

A second key observation is that imposing non-positive constraints on two adjacent layers with an activation function in between is equivalent to imposing non-negative constraints on the two layers while replacing the activation with its point reflection.

Proposition 10.7.3. *An MLP with $W^{(k)} \leq 0$, $W^{(k+1)} \leq 0$ and $\sigma^{(k)}(x) = \sigma(x)$ is equivalent to an MLP with $W^{(k)} \geq 0$, $W^{(k+1)} \geq 0$ and $\sigma^{(k)}(x) = \sigma'(x) = -\sigma(-x)$.*

It follows that an MLP with an even number of layers, non-positive weights, and activation σ at all layers is equivalent to an MLP with non-negative weights and alternating activations between σ and σ' . This equivalence can be obtained via proposition 10.7.3 by flipping the weight constraints two layers at a time, thereby changing the activations at odd-numbered layers from σ to σ' .

Thanks to proposition 10.7.3, we immediately obtain the following:

Proposition 10.7.4. *If $\sigma \in \mathcal{S}^- \cup \mathcal{S}^+$, an MLP with 4 layers, non-positive weights, and activation σ is a universal approximator for the class of monotonic functions.*

These observations also imply that the structure proposed in [RS23b] can yield universal monotonic approximators using only point reflections, without requiring the third activation class.

While theorem 10.6.5 establishes that a broad class of constrained MLP architectures are universal monotonic approximators, this does not guarantee ease of optimization. Consider an MLP with ReLU activations and biases initialized to zero. For an input $x \geq 0$, we have $-|W|x \leq 0$, and thus the first layer output is $\text{ReLU}(-|W|x) = 0$. Conversely, if $x \leq 0$, then $\text{ReLU}(-|W|x) \geq 0$, causing the second layer to saturate instead. To enable efficient training, bias initialization must be carefully tuned to avoid zero gradients throughout. More generally, weight constraints make these networks more sensitive to initialization.

10.8 Relaxing weight constraints

Historically, the earliest works on monotonic neural networks relied on constraining the network parameters to be non-negative. In particular, enforcing non-negativity on the weight matrices W in the affine transformations, combined with bounded monotonic activations, is a sufficient condition to guarantee that the overall function is monotonic [DV10, SAM96]. More recently, [RS23b] demonstrated how to build effective monotonic MLPs using this approach by employing multiple activations.

Although constrained weights with bounded activations are straightforward to implement and compatible with standard gradient-based optimizers, poor initialization can still lead to vanishing gradient dynamics, as further discussed in section 10.11 and section 10.10.1. In this work, we propose an alternative formulation that addresses this issue while also removing the need to alternate activation saturation to achieve universal approximation.

If we adopt the weight-constrained formulation for the construction in section 10.7, we are still left with the problem of selecting the activation sequence for the MLP, which may be unclear or require additional hyperparameter tuning. However, by slightly rearranging the order of operations, we can design a monotonic MLP that eliminates manual tuning of the saturation side while simultaneously relaxing the weight constraint.

Consider a single layer $f(x) = \sigma(|W|x + b)$ in a constrained MLP, where

absolute value is used for weight reparametrization. Instead of constraining weights, we can decompose W into its positive and negative parts, $W^+ = \max(W, 0)$ and $W^- = \min(W, 0)$. The affine transformation can then be written as

$$|W|x + b = W^+x - W^-x + b. \quad (10.15)$$

Applying the non-linearity to each term of eq. (10.15) individually, while sharing the bias term, yields the parametrization

$$\hat{f}(x) = \sigma(W^+x + b) - \sigma(W^-x + b). \quad (10.16)$$

Proposition 10.8.1. *Any function representable using an affine transformation with non-negative weights followed by either σ or σ' can also be represented using eq. (10.16), up to a constant factor.*

Proof. When all entries of W share the same sign, one of the two terms in eq. (10.16) reduces to $\pm\sigma(b)$. Specifically, if $W \geq 0$, the expression simplifies to $\sigma(|W|x + b) - \sigma(b)$, while if $W \leq 0$, it becomes $\sigma(b) - \sigma(-|W|x + b)$. To conclude, recall that $-\sigma(-x) = \sigma'(x)$. $\square$

The additional constant factor can be absorbed into the bias term of the subsequent layer. Therefore, proposition 10.8.1 encompasses both cases in proposition 10.7.2. This implies that an MLP constructed by stacking at least four blocks parameterized as in eq. (10.16) is a universal approximator for monotonic functions. Hence, the proposed formulation is strictly more expressive than a simple weight-constrained approach, which arises as a special case of eq. (10.16).

Alternatively, one can apply the same reasoning in reverse, starting from the last layer of the network, leading to an alternative formulation:

$$\hat{f}(x) = W^+\sigma(x) + W^-\sigma(-x) + b. \quad (10.17)$$

We refer to eq. (10.16) and eq. (10.17) as the pre-activation switch and post-activation switch, respectively.

We present both the pseudocode and the computational graph for the post-activation formulation. For completeness, this section also includes the pre-activation pseudocode and computational graph. To improve readability and facilitate direct comparison, we report them side by side: the two computational graphs are shown in fig. 10.6, while the corresponding pseudocodes are listed in section 10.8.

Algorithm 3 Forward pass of a Monotonic ReLU MLP with pre-activation switch

Input: data $x \in \mathbb{R}^n$, weight matrix $W \in \mathbb{R}^{h_l \times h_{l-1}}$, bias vectors $b \in \mathbb{R}^{h_l}$, activation function σ

Output: prediction $\hat{y} \in \mathbb{R}^{h_L}$

$W^+ := \max(W, 0)$

$W^- := \min(W, 0)$

$z^+ := W^+x + b$

$z^- := W^-x + b$

$\hat{y} := \sigma(z^+) - \sigma(z^-)$

Algorithm 4 Forward pass of a Monotonic ReLU MLP with post-activation switch

Input: data $x \in \mathbb{R}^n$, weight matrix $W \in \mathbb{R}^{h_l \times h_{l-1}}$, bias vectors $b \in \mathbb{R}^{h_l}$, activation function σ

Output: prediction $\hat{y} \in \mathbb{R}^{h_L}$

$W^+ := \max(W, 0)$

$W^- := \min(W, 0)$

$z^+ := W^+\sigma(x)$

$z^- := W^-\sigma(-x)$

$\hat{y} := z^+ + z^- + b$

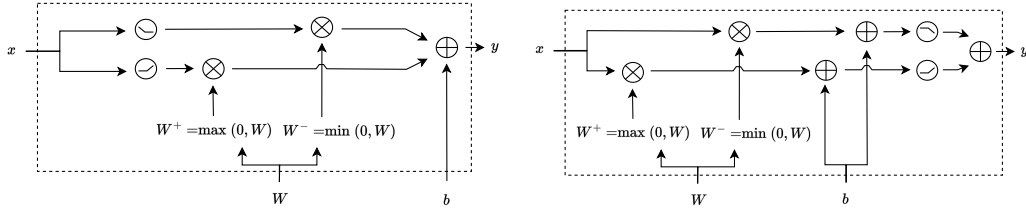


Figure 10.6: Computation graph of a single layer of a ReLU monotonic NN with the proposed learned activation via weight sign. The left plot reports the computational graph of the post-activation, and the right plot shows the pre-activation switch.

The simplicity of this approach is evident: it preserves the structure of the forward pass of a traditional MLP, thanks to the relaxation of the weight constraint, and requires no special care for initialization. Further details on its optimization properties are discussed in section 10.11. For all experiments, we used the default PyTorch initialization [PGM⁺19] without additional tuning. A naive implementation requires only an additional matrix multiplication. This operation can be easily parallelized and incurs no measurable computational overhead in our experiments.

10.9 Extension to other activations

In all practical examples we assume ReLU as the activation function for the MLP. However, the results in section 10.7 and section 10.8 require only that the activation function be monotonic and saturate on at least one side. While ReLU satisfies these conditions, it is not unique in this respect; many other widely used ReLU-like activations also meet the minimal assumptions of theorem 10.6.5. For this reason, we analyze a broader set of activations and report their compatibility with our construction. Specifically, in table 10.1 we summarize several widely adopted activations, along with their gradients, whether they are non-decreasing and saturating, and whether they can be applied within the proposed framework.

The results show that the proposed method is compatible with most commonly used activations today. Importantly, while the method also supports bounded activations such as sigmoid and tanh, these offer little additional benefit over the weight-constrained formulation. In contrast, the real advantage arises with activations that are monotonic and saturate on only one side, as these fully exploit the flexibility of the proposed approach.

Table 10.1: Widely used activations with their corresponding properties, and whether they can be used or not.

Name	Function	Gradient	Monotone	Saturates	Usable
ReLU	$\begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases}$	$\begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases}$	✓	✓	✓
LeakyReLU	$\begin{cases} x & \text{if } x \geq 0 \\ \alpha x & \text{otherwise} \end{cases}$	$\begin{cases} 1 & \text{if } x \geq 0 \\ \alpha & \text{otherwise} \end{cases}$	✓ ¹	✗	✗
PReLU	$\begin{cases} x & \text{if } x \geq 0 \\ \alpha x & \text{otherwise} \end{cases}$ (α learnable)	$\begin{cases} 1 & \text{if } x \geq 0 \\ \alpha & \text{otherwise} \end{cases}$	✓ ¹	✓	✓ ¹
ReLU6	$\begin{cases} 6 & \text{if } x \geq 6 \\ x & \text{if } 0 \leq x \leq 6 \\ 0 & \text{otherwise} \end{cases}$	$\begin{cases} 0 & \text{if } x \geq 6 \\ 1 & \text{if } 0 \leq x \leq 6 \\ 0 & \text{otherwise} \end{cases}$	✓	✓	✓
ELU	$\begin{cases} x & \text{if } x \geq 0 \\ \alpha(e^x - 1) & \text{otherwise} \end{cases}$	$\begin{cases} 1 & \text{if } x \geq 0 \\ \alpha e^x & \text{otherwise} \end{cases}$	✓ ¹	✓	✓ ¹
SELU	$\lambda \begin{cases} x & \text{if } x \geq 0 \\ \alpha(e^x - 1) & \text{otherwise} \end{cases}$	$\lambda \begin{cases} 1 & \text{if } x \geq 0 \\ \alpha e^x & \text{otherwise} \end{cases}$	✓ ¹	✓	✓ ¹
CELU	$\lambda \begin{cases} x & \text{if } x \geq 0 \\ \alpha(\frac{e^x}{\alpha} - 1) & \text{otherwise} \end{cases}$	$\lambda \begin{cases} 1 & \text{if } x \geq 0 \\ \frac{e^x}{\alpha} & \text{otherwise} \end{cases}$	✓ ¹	✓	✓ ¹
GeLU	$x\Phi(x)$	$\Phi(x)\frac{1}{\sqrt{2\pi}}e^{\frac{-x^2}{2}}$	✗	✓	✗
SiLU/Swish	$x\sigma(x)$	$\frac{e^x(x+e^x+1)}{(e^x+1)^2}$	✗	✓	✗
Sigmoid	$\frac{1}{1+e^{-x}}$	$\frac{e^{-x}}{(1+e^{-x})^2}$	✓	✓	✓
Tanh	$\frac{e^x - e^{-x}}{e^x + e^{-x}}$	$1 - \left(\frac{e^x - e^{-x}}{e^x + e^{-x}}\right)^2$	✓	✓	✓
Exp	e^x	e^x	✓	✓	✓
SoftSign	$\frac{x}{ x +1}$	$\frac{1}{(x +1)^2}$	✓	✓	✓
Softplus	$\log(1 + e^x)$	$\frac{e^x}{e^x+1}$	✓	✓	✓
LogSigmoid	$-\log(1 + e^{-x})$	$\frac{1}{1+e^x}$	✓	✓	✓

¹: true only if parametrized in such a way to guarantee $\alpha \geq 0$

10.10 Experimental results

10.10.1 Comparison of different parametrizations on toy datasets

To highlight the effectiveness of the proposed method compared to the bounded-activation counterpart, we present a simple synthetic example in fig. 10.7. Specifically, the models are tasked with approximating $f(x) = \cos(x) + x$, a one-dimensional monotonic function with multiple saddle points. This choice requires the approximation model to be highly flexible. We compare four different models:

- Unconstrained NN: serves as a baseline, showing that an unconstrained model can learn the target function.
- Monotonic NN with non-negative weights and ReLU activations: illustrates the theoretical limitation that such models cannot approximate non-convex functions.
- Monotonic NN with non-negative weights and sigmoid activations: a universal approximator for monotonic functions but limited by vanishing gradients.
- Proposed parametrization (post-activation switch, section 10.8): our method, designed to overcome these limitations.

As shown in fig. 10.7, the monotonic model with non-negative ReLU activations fails to learn the function, as predicted by theory, since the target is non-convex. By contrast, both the sigmoid model and our proposed approach successfully approximate it. However, the sigmoid model suffers from optimization difficulties due to vanishing gradients. In contrast, our method leverages rectified linear activations, which are much easier to optimize when the number of dead neurons remains manageable, as explained in [GB10, RPK⁺17].

This difference is further evident when analyzing the Negative Log-Likelihood (NLL) loss during training. In fig. 10.7, we report the training losses for two different layer widths. As expected, the naive monotonic ReLU model performs worst, being unable to approximate the function. The sigmoid-based monotonic NN, despite its universal approximation guarantee, learns more slowly, likely due to vanishing gradients. By contrast, the proposed method

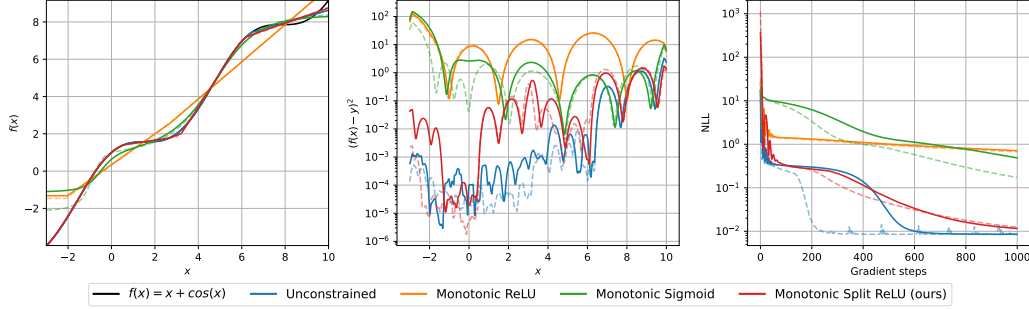


Figure 10.7: First plot, approximation of $f(x)$ using MLPs with layers of 128 neurons. Second plot, approximation of $f(x)$ using MLPs with layers of 256 neurons. Last plot, training losses of the different methods (full lines represent versions with 128 neurons, dashed lines represent versions with 256 neurons).

with ReLU activations converges fastest, nearly matching the unconstrained model in the larger-width setting.

As discussed at the end of section 10.7, MLPs with constrained weights require careful initialization to avoid non-optimizable configurations. The proposed method in section 10.8 alleviates this issue, though it is not entirely immune to it.

To illustrate the vanishing gradient problem exacerbated by non-negative constraints, in fig. 10.8 we construct a 128-neuron wide MLP with varying numbers of hidden layers and compare the average gradient of the output with respect to the parameters on the same function approximation problem presented in fig. 10.7. The results show that the sigmoid-based monotonic MLP has much smaller gradients, even for shallow networks: its average gradient is 0.0019 for 4 layers and 0.00099 for 10 layers. By contrast, the monotonic ReLU MLP exhibits exploding gradients due to the accumulation of activations from the combination of ReLU and non-negative weights, starting from 3.54 for 4 layers and reaching 3311.00 for 10 layers. The proposed method, however, maintains gradients within a reasonable range, from 0.010 at 4 layers to 1.259 at 10 layers. All results are averaged over 20 random initializations, and the plots show $\pm 1\sigma$.

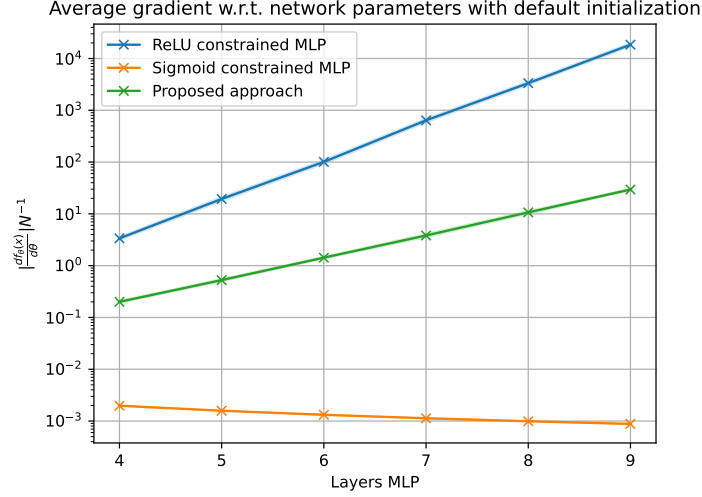


Figure 10.8: Average gradient from monotonic MLPs varying the number of layers. Data is shown in the log scale for the y -axis.

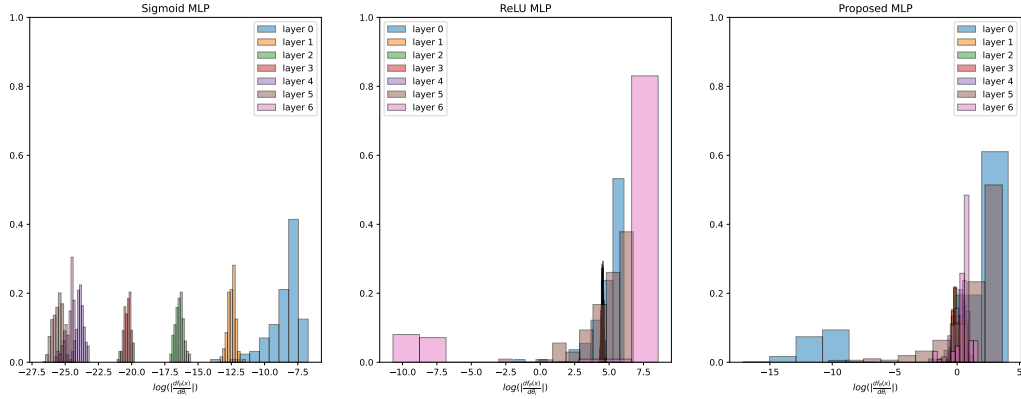


Figure 10.9: Distribution of gradients from monotonic MLPs for each layer (layer 0 is the final one, layer 6 is the first after the input).

To further analyze these optimization challenges, fig. 10.9 reports the gradient distributions of 6-layer MLPs under the three architectures. The sigmoid MLP displays extremely small gradients in the initial layers, leading to slow learning, while the ReLU MLP shows exploding gradients in the

deeper layers. The proposed method avoids both extremes.

Finally, it is worth noting that using ReLU6 under the same configuration produces a dead network, as all gradients vanish due to the saturation of the activation.

10.10.2 Comparison on Real-world Datasets

In this section, we evaluate the performance of the proposed method against alternative approaches that provide monotonicity guarantees. To ensure consistency with prior work, our code was heavily based on the implementation by [RS23b], which also determines the datasets used. For completeness, we briefly describe the datasets here; for a more detailed account, see [RS23b].

- **COMPAS** [FMSS22]: Binary classification dataset of criminal records with 13 features, 4 of which are monotonic.
- **Blog Feedback** [Buz13]: Regression dataset with 280 features, 8 of which are monotonic, used to predict the number of comments within 24 hours.
- **Auto MPG**: Regression dataset with 7 features, 3 of which are monotonic, aimed at predicting miles-per-gallon consumption.
- **Heart Disease**: Binary classification dataset with 13 features, 2 of which are monotonic, used to predict the presence of heart disease.
- **Loan Defaulter**: Binary classification dataset with 28 features, 5 of which are monotonic, aimed at predicting loan defaults.

Among the classification datasets, COMPAS includes 13 features, 4 of which exhibit monotonic relationships with the label. Heart Disease contains 13 features with 2 monotonic covariates, while Loan Defaulter includes 28 features with 5 monotonic dependencies. For regression tasks, Auto MPG has 7 features, 3 of which are monotonically decreasing with respect to the output. Finally, Blog Feedback differs from the others in that only a small fraction of its covariates are monotonic: out of 280 features, only 8 are monotonic, representing 2.8% of the total.

We compare our method with several alternative approaches that provide monotonic guarantees by construction. Specifically, we include XGBoost [CG16], Deep Lattice Networks [YDC⁺17], Min-Max Networks [DV10], Certified Networks [LHZL20], COMET [SFMVdB20], Constrained Monotonic

Neural Networks [RS23b], Expressive Monotonic Neural Networks [NKW23], and Scalable Monotonic Neural Networks [KL24]. We also report results for a naive constrained monotonic MLP using sigmoid activations, denoted Non-Neg-DNN. Similar baselines were presented in [LHZL20], though on a more restricted set of datasets. The final test set metrics are summarized in table 10.2.

For the proposed method, little to no hyperparameter tuning was required, since the hyperparameters from [RS23b] transferred directly without adjustment. Despite this, the proposed method consistently matches or exceeds the performance of all other recently introduced approaches.

The specifications used to obtain the results in table 10.3 are as follows. All experiments were implemented in PyTorch and trained with the Adam optimizer from the PyTorch library. For all datasets except Blog Feedback, we reused networks with similar size and architecture to those in [RS23b], without further tuning. For Blog Feedback, however, limited hyperparameter tuning was necessary due to the datasets large size, sparse features, and small proportion of monotonic covariates (only about 3%). In this case, CELU activations were used for smaller MLPs to mitigate the risk of dead neurons. Code available at github.com/AMCO-UniPD/monotonic

Table 10.2: Test metrics across different datasets. The best-performing architecture per dataset is bolded.

Method	COMPAS (Test Accuracy)	Blog Feedback (Test RMSE)	Loan Defaulter (Test Accuracy)	AutoMPG (Test MSE)	Heart Disease (Test Accuracy)
XGBoost	68.5% $\pm$ 0.1%	0.176 $\pm$ 0.005	63.7% $\pm$ 0.1%	-	-
Certified	68.8% $\pm$ 0.2%	0.159 $\pm$ 0.001	65.2% $\pm$ 0.1%	-	-
Non-Neg-DNN	69.3% $\pm$ 0.1%	0.154 $\pm$ 0.001	65.2% $\pm$ 0.1%	10.31 $\pm$ 1.86	89% $\pm$ 1%
DLN	67.9% $\pm$ 0.3%	0.161 $\pm$ 0.001	65.1% $\pm$ 0.2%	13.34 $\pm$ 2.42	86% $\pm$ 2%
Min-Max Net	67.8% $\pm$ 0.1%	0.163 $\pm$ 0.001	64.9% $\pm$ 0.1%	10.14 $\pm$ 1.54	75% $\pm$ 4%
Constrained MNN	69.2% $\pm$ 0.2%	0.154 $\pm$ 0.001	65.3% $\pm$ 0.1%	8.37 $\pm$ 0.08	89% $\pm$ 0%
Scalable MNN	69.3% $\pm$ 0.9%	0.150 $\pm$ 0.001	65.0% $\pm$ 0.1%	7.44 $\pm$ 1.20	88% $\pm$ 4%
Expressive MNN	69.3% $\pm$ 0.1%	0.160 $\pm$ 0.001	65.4% $\pm$ 0.1%	7.58 $\pm$ 1.20	90% $\pm$ 2%
Ours	69.5% $\pm$ 0.1%	0.149 $\pm$ 0.001	65.4% $\pm$ 0.1%	7.34 $\pm$ 0.46	94% $\pm$ 1%

Table 10.3: Hyper-parameters used for results reported in table 10.2

Hyper-parameter	COMPAS	Blog Feedback	Loan Defaulter	AutoMPG	Heart Disease
Learning-rate	10^{-3}	10^{-2}	10^{-3}	10^{-3}	10^{-3}
Epochs	100	1000	50	300	300
Batch-size	8	256	256	8	8
Free layers size	16	2	16	8	16
Number of free layers	3	2	3	3	3
Monotonic layers size	16	3	16	8	16
Number of monotonic layers	3	2	3	3	3
Activation	ReLU	CELU	ReLU	CELU	ReLU

10.11 Initialization Vanishing Gradient in Constrained MLPs

As discussed in section 10.3, a naive way to ensure monotonicity is to use monotonic activations together with non-negative weight constraints. In this case, the affine transformations are typically parametrized as $l(x) = g(W)x + b$, for some mapping $g : \mathbb{R} \rightarrow \mathbb{R}_+$. Note that the bias can take any value, since being constant it does not affect the gradient.

Such networks often employ bounded activations (such as sigmoid, tanh, or ReLU6) to preserve convex-concave properties. However, this design makes them highly sensitive to initialization and prone to vanishing gradients [GB10]. To see why constraining weights exacerbates this issue, consider a monotonic MLP with sigmoidal activations initialized with standard schemes (e.g., Glorot), where weights are sampled from a symmetric distribution and biases are set to zero. Using $g(x) = |x|$ (though the same reasoning applies to other g), each layer takes the form $\sigma(x) = \sigma(|W|x + b)$. After the first layer, $\sigma^{(1)}(x) \in (0, 1)$, so in the second layer $|W^{(2)}|\sigma^{(1)}(x)$ is always non-negative and often large. Passing this through a sigmoid leads to saturation, yielding near-zero gradients. This effect compounds with depth, worsening for deeper networks. Similar issues arise for ReLU6 MLPs (where gradients may vanish entirely) and for tanh MLPs (particularly on normalized datasets, which is standard preprocessing).

Clear evidence of this phenomenon appears in section 10.10.1, where even a simple 1D task shows vanishing gradients in constrained sigmoidal MLPs after only a few layers (fig. 10.7). Moreover, signs of vanishing gradients are visible already at initialization, as shown in fig. 10.8.

One possible remedy is the inclusion of Batch Normalization layers [Iof15], which are known to alleviate initialization and optimization issues. Batch-Norm applies the transformation

$$BN(x) = \frac{x - \mathbb{E}[x]}{\sqrt{\text{Var}[x] + \epsilon}} \cdot \gamma + \beta$$

Since $\sqrt{\text{Var}[x] + \epsilon} > 0$, constraining $\gamma \geq 0$ (e.g., via $\gamma = \text{SoftPlus}(\gamma')$) ensures monotonicity. BatchNorm is typically initialized with $\beta = 0$ and $\gamma = 1$. When placed before activations, it could prevent exploding pre-activations by standardizing them around zero. However, investigating this approach lies beyond the scope of this work and is left as a future direction.

The proposed activation switch parametrization (eq. (10.16), eq. (10.17)) addresses these limitations by removing the explicit weight constraints needed to guarantee monotonicity, while also eliminating the need to manually alternate activations. As discussed earlier, weight constraints combined with bounded activations often lead to vanishing gradients, with ReLU6 in particular producing almost entirely dead networks (section 10.10.1). Our parametrization mitigates this problem in two ways: avoiding explicit weight constraints and enabling the use of unbounded activations.

Nevertheless, the approach still departs from the classical MLP formulation. In particular, splitting the weight matrix into positive and negative parts may still introduce challenges related to initialization.

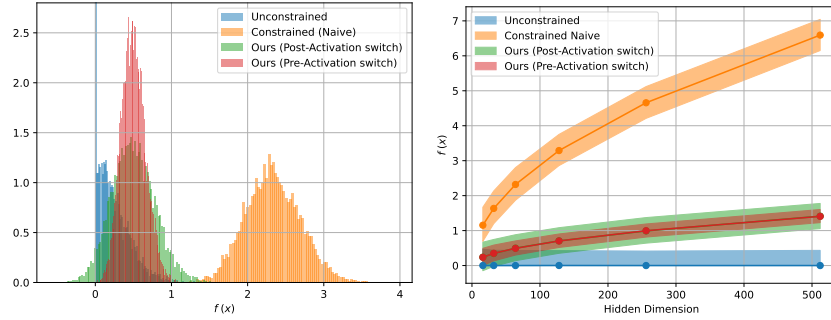


Figure 10.10: First plot, the distribution of the output of an MLP with the different parametrizations. Second plot, the scaling law of the expected output after initialization of the different parametrization. In both images, it can be seen how the naive constrained MLP has a very different scaling behavior compared to the rest.

From an empirical perspective, fig. 10.10 shows the distribution of outputs from multilayer MLPs with different parametrizations. Unconstrained refers to a standard MLP, Constrained Naive to an MLP with $|W|$ weight parametrization, and Pre/Post Activation Switch to the formulations in eq. (10.16) and eq. (10.17). As expected, the unconstrained MLP produces outputs with zero mean regardless of hidden dimension. In contrast, the activation-switch formulations exhibit a slight upward drift in the expected output under random initialization, consistent with theoretical predictions. However, this increase is far smaller than that induced by naively constrain-

ing weights to be positive.

Empirically, the activation-switch parametrization does not suffer from initialization issues. The results reported in table 10.2 were obtained using the default PyTorch initialization, and aggregated over multiple random seeds. Hence, the networks were consistently initialized with different values yet still produced stable outcomes.

Overall, while the behavior of activation-switch parametrizations slightly deviates from that of standard MLPs, this difference has no adverse impact on performance. Moreover, the deviation remains significantly smaller than that observed in the naive weight-constrained counterpart without activation switching.

10.12 Conclusions

In this work, we relaxed the requirements for achieving universal approximation in monotonic MLPs with constrained weights. We proved that alternating the saturation side of the activations is sufficient to guarantee universal approximation with a finite number of layers. Furthermore, we established a connection between the saturation side of the activations and the sign of the weight constraint. This led to the surprising result that non-positive weight constraints are more expressive than non-negative ones, which can only represent convex functions.

Building on this theoretical analysis, we introduced a novel parameterization that relaxes the weight constraint, thereby reducing sensitivity to initialization. In our formulation, the activation saturation side is learnable, ensuring universal approximation even with monotonic convex activations, a capability not possible in prior approaches. Empirically, MLPs constructed with our fully connected monotone layer achieve state-of-the-art performance.

While our results demonstrate that any monotonic saturating activation can be used to build monotonic MLPs, it remains an open question whether non-saturating activations, such as Leaky-ReLU, can also provide universal approximation under monotonicity constraints. In addition, although batch normalization has shown strong effectiveness in unconstrained settings, its potential as a solution to the initialization problem in the monotonic case has yet to be explored.

11

Conclusions and Discussion

The central aim of this thesis was to explore how machine learning systems can be made more trustworthy by addressing their reliability, transparency, and adaptability. While machine learning has transformed countless domains, the widespread deployment of these systems in safety-critical, regulated, or ethically sensitive contexts requires properties beyond predictive accuracy. Models must also be robust, interpretable, and fair.

This thesis has approached these challenges by investigating three complementary perspectives:

- **Anomaly Detection with Human Feedback:** developing mechanisms to align machine learning models with expert knowledge in dynamic and uncertain environments.
- **Safe and Interpretable Reinforcement Learning:** ensuring that decision-making processes operate under safety constraints and are understandable by human operators.
- **Encoding Desirable Properties into Models:** designing methods for representation learning and neural network architectures that embed fairness and monotonicity guarantees.

Together, these contributions serve to advance Trustworthy AI, offering theoretical insights but also practical algorithms.

11.0.1 Anomaly Detection with Human Feedback

The contributions of this thesis in the first area have been: An extension of the widely used Isolation Forest by incorporating expert feedback through efficient update rules. A Bayesian framework that treats the model update probabilistically, enabling principled uncertainty estimation and more informed querying strategies. The extension of the framework to other tree ensembles methods and an efficient strategy for performing updates in batches, improving usability and scalability.

There remains several limitations and possible avenues for future work. While BALEIF proved that the update mechanism of BALIF and ALIF can be effectively extended to other tree based detector that have a more desirable space partitioning, it remains to be seen how far from the original IF model this concept can be pushed. In particular, it would be interesting to extend the framework to more flexible or dynamic tree structures (e.g., Mondrian Forests) that adapt to evolving distributions.

This would require the relaxation of the tight coupling between updates and fixed partitions, and modelling of intra-region interactions and dependencies across tree regions. This could be achieved by assigning extra model parameters to internal nodes, similarly to what has been done for the leafs. The extension to dynamic models would also enable the model to adapt to data drift, i.e. to a changing feature distribution, rather than only tackling the issue of concept drift in the definition of anomaly.

Moreover, user-provided labels have been assumed to be correct, but real-world annotations are often noisy or inconsistent. Future work in this direction would include the development of a mechanisms to incorporate uncertain or erroneous feedback, such as confidence-weighted annotations. In this case, the user could provide a confidence level associated to its feedback, and the model update could be regulated accordingly.

Overall, this area of research offers promising initial results and the focus on the user experience and usability of the system can aid in its adoption by professionals in the field.

11.0.2 Safe and Interpretable Reinforcement Learning

The contributions of this thesis in the second area have been: Introducing constrained policy optimization strategies for active fault diagnosis, ensur-

ing safe exploration and/or adherence to operational constraints. We also proposed a method for generating structured options that allow RL agents to operate at multiple levels of abstraction, improving both interpretability and efficiency.

The MANGO framework and the FIERL-based CRL passive diagnosis approach both demonstrate significant advances in reinforcement learning for complex and uncertain environments, yet, while ambitious, they face notable limitations. MANGO’s performance is critically constrained by the reliability of lower-layer policies, sensitivity to training schedules, and reliance on hand-crafted abstractions, which may not generalize to generic and unstructured domains. Training layer sequentially starting from low-level abstraction also allows early-layer errors to propagate upward, which end up affecting higher-level policies both during training and during deployment. Future work aims to address these issues through end-to-end joint optimization of all abstraction layers. Furthermore the development of a mechanism for automatic hierarchy and abstraction discovery will be critical in extending the framework to more complex and realistic environments.

Similarly, the CRL based approach FIERL finds limited application due to the instability and computational demands of RL, that are exacerbated by the constrained learning approach. This limitations, unfortunately, are critical in most domains of interest. In particular, difficulty in assessing convergence across the full state space and reliance on accurate system models might by themselves be enough to distance practitioners from the approach. Additionally, the current cost formulation can lead to objective misalignment, allowing localized violations despite satisfying global expectations. Future directions include refining cost designs to penalize repeated or concentrated violations and extending the framework to non-linear, over-actuated, or more general stochastic fault scenarios. Collectively, these extensions aim to enhance scalability, robustness, and practical applicability in real-world control systems.

11.0.3 Encoding Desirable Properties into Models

The contributions of this thesis in the third area have been: Introducing simplified penalties that guarantee independence between learned representations and sensitive attributes, reducing computational cost while maintaining performance. Furthermore, we proposed novel architectures with provable universal approximation properties under monotonicity constraints, and demonstrated improved performance and stability at initialization.

Both contributions advance their respective fields, yet, once again face notable limitations. The CFD approach struggles to enforce the fairness constraint in high-dimensional settings, and the current fairness benchmarks, being mostly low-dimensional, limit evaluation under realistic conditions. Future work should focus on improving the scalability and robustness of CFD-based methods, either via improved sampling in the transformed domain, or by learning the kernel parameters via the incorporation of limited adversarial learning. This should be accompanied by the development of richer, high-dimensional fairness benchmarks to better assess performance and fairness trade-offs.

Similarly, while monotonic MLPs with constrained weights achieve universal approximation with saturating activations, it remains unclear whether non-saturating activations, such as Leaky-ReLU, can also provide universal approximation under monotonicity constraints. This work was mainly focused on the theoretical aspects of the universal approximation properties of such networks, and an further empirical studies are needed to test monotonic neural networks in domains such as finance, law, and healthcare, where monotonicity constraints are required, be it for ethical and practical reasons or legally mandated by regulations. Additionally, the potential of batch normalization to mitigate sensitivity to initialization in monotone networks has yet to be explored. Investigating non-saturating activations and techniques like batch normalization could enhance robustness and practical applicability of monotonic MLPs, inching closer to the unconstrained architectures that are used in practice.

11.0.4 Final Discussion

This thesis has several broader implications for the design and deployment of machine learning systems. Trustworthy AI cannot rely solely on unsupervised or fully automated methods. Incorporating domain expertise, even in small amounts, substantially improves alignment with real-world requirements.

While this thesis provides meaningful advances, it also opens several avenues for further research. In conclusion, this thesis has contributed to the emerging field of Trustworthy AI by advancing anomaly detection, safe reinforcement learning, and fair representation learning. The proposed methods demonstrate that trustworthiness is not only a theoretical aspiration but a practical design principle. This work represents a step toward a future where machine learning models are not only powerful and accurate but also accountable, fair, robust, and interpretable.

Bibliography

- [AAO17] John O Awoyemi, Adebayo O Adetunmbi, and Samuel A Oluwadare. Credit card fraud detection using machine learning techniques: A comparative analysis. In *2017 international conference on computing networking and informatics (ICCNi)*, pages 1–9. IEEE, 2017.
- [Abe22] David Abel. A theory of abstraction in reinforcement learning. In *arXiv preprint arXiv:2203.00397*, 2022.
- [AHTA17] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *ICML*, pages 22–31, 2017.
- [Alt99] E. Altman. *Constrained Markov Decision Processes*. Chapman and Hall, 1999.
- [AMN⁺19] Angelos Angelopoulos, Emmanouel T Michailidis, Nikolaos Nomikos, Panagiotis Trakadas, Antonis Hatziefremidis, Stamatis Voliotis, and Theodore Zahariadis. Tackling faults in the industry 4.0 eraa survey of machine-learning solutions and key aspects. *Sensors*, 20(1):109, 2019.
- [ASS20] Abdul Fatir Ansari, Jonathan Scarlett, and Harold Soh. A characteristic function approach to deep implicit generative modeling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7478–7487, 2020.
- [AUK⁺20] David Abel, Nathan Umbanhowar, Khimya Khetarpal, Dilip Arumugam, Doina Precup, and Michael L. Littman. Value preserving state-action abstractions. In *International Conference on Artificial Intelligence and Statistics*, 2020.

- [Bar12] David Barber. *Bayesian reasoning and machine learning*. Cambridge University Press, 2012.
- [Bar17] Jonathan T Barron. Continuously differentiable exponential linear units. *arXiv preprint arXiv:1704.07483*, 2017.
- [BCMS22] Tommaso Barbariol, Filippo Dalla Chiara, Davide Marcato, and Gian Antonio Susto. A review of tree-based approaches for anomaly detection. *Control Charts and Machine Learning for Anomaly Detection in Manufacturing*, pages 149–185, 2022.
- [BDE09] Tim Brennan, William Dieterich, and Beate Ehret. Evaluating the predictive validity of the compas risk and needs assessment system. *Criminal Justice and Behavior*, 2009.
- [BFS20] Tommaso Barbariol, Enrico Feltresi, and Gian Antonio Susto. Self-diagnosis of multiphase flow meters through machine learning-based anomaly detection. *Energies*, 13(12):3136, 2020.
- [BHN23] Solon Barocas, Moritz Hardt, and Arvind Narayanan. *Fairness and Machine Learning: Limitations and Opportunities*. MIT Press, 2023.
- [BKNS00] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.
- [Bla24] Harrison Blake. Algorithmic fairness: Developing methods to detect and mitigate bias in ai systems. 11 2024.
- [BRV22] Mislav Balunovic, Anian Ruoss, and Martin Vechev. Fair normalizing flows. In *International Conference on Learning Representations*, 2022.
- [BSS⁺24] Dariusz Brzezinski, Julia Stachowiak, Jerzy Stefanowski, Izabela Szczech, Robert Susmaga, Sofya Aksenjuk, Uladzimir Ivashka, and Oleksandr Yasinskyi. Properties of fairness measures in the context of varying class imbalance and protected

- group ratios. *ACM Transactions on Knowledge Discovery from Data*, 2024.
- [Buz13] Krisztian Buza. Feedback prediction for blogs. In *Data analysis, machine learning and knowledge discovery*, pages 145–152. Springer, 2013.
- [CAB⁺23] Giandomenico Cornacchia, Vito Walter Anelli, Giovanni Maria Biancofiore, Fedelucio Narducci, Claudio Pomo, Azzurra Ragone, and Eugenio Di Sciascio. Auditing fairness under unawareness through counterfactual reasoning. *Inf. Process. Manag.*, 60(2):103224, 2023.
- [CBK09] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41, 07 2009.
- [CG16] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.
- [CGJP17] Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Res.*, 18(1):6070–6120, 2017.
- [CHA⁺23] Abigail Copiaco, Yassine Himeur, Abbes Amira, Wathiq Mansoor, Fodil Fadli, Shadi Atalla, and Shahab Saquib Sohail. An innovative deep anomaly detection of building energy consumption using energy time-series images. *Engineering Applications of Artificial Intelligence*, 119:105775, 2023.
- [Cho17] Alexandra Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data*, 5(2):153–163, 2017.
- [CLC⁺21] Fabrizio Carcillo, Yann-Al Le Borgne, Olivier Caelen, Yacine Kessaci, Frdrick Obl, and Gianluca Bontempi. Combining unsupervised and supervised learning in credit card fraud detection. *Information Sciences*, 557:317–331, 2021.

- [CLC⁺24] A. Feder Cooper, Katherine Lee, Madiha Zahrah Choksi, Solon Barocas, Christopher De Sa, James Grimmelmann, Jon M. Kleinberg, Siddhartha Sen, and Baobao Zhang. Arbitrariness and social prediction: The confounding role of variance in fair classification. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan, editors, *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2024, February 20-27, 2024, Vancouver, Canada*, pages 22004–22012. AAAI Press, 2024.
- [CMBS19] Mattia Carletti, Chiara Masiero, Alessandro Beghi, and Gian Antonio Susto. A deep learning approach for anomaly detection with industrial time series data: a refrigerators manufacturing case study. *Procedia Manufacturing*, 38:233–240, 2019.
- [CMF20] Andrew A. Cook, Gksel Msrl, and Zhong Fan. Anomaly detection for iot time-series data: A survey. *IEEE Internet of Things Journal*, 7(7):6481–6494, 2020.
- [CMJ⁺19] Elliot Creager, David Madras, Jörn-Henrik Jacobsen, Marissa Weis, Kevin Swersky, Toniann Pitassi, and Richard Zemel. Flexibly fair representation learning by disentanglement. In *International conference on machine learning*, pages 1436–1445. PMLR, 2019.
- [CMS⁺02] Carmen Carroll, Phil Marsden, Pat Soden, Emma Naylor, John New, and Tim Dornan. Involving users in the design and usability evaluation of a clinical decision support system. *Computer Methods and Programs in Biomedicine*, 69(2):123–135, 2002.
- [Cou14] Council of European Union. Council regulation (EU) no 269/2014, 2014.
<http://eur-lex.europa.eu/legal-content/EN/TXT/?qid=1416170084502&uri=CELEX:32014R0269>.

- [CRSG15] Kacper P Chwialkowski, Aaditya Ramdas, Dino Sejdinovic, and Arthur Gretton. Fast two-sample testing with analytic representations of probability measures. *Advances in Neural Information Processing Systems*, 28, 2015.
- [CS20] Pawel Chilinski and Ricardo Silva. Neural likelihoods via cumulative distribution functions. In *Conference on Uncertainty in Artificial Intelligence*, pages 420–429. PMLR, 2020.
- [CTC⁺22] Dongyue Chen, Xinyue Tantai, Xingya Chang, Miaoting Tian, and Tong Jia. Weakly supervised anomaly detection based on two-step cyclic iterative pu learning strategy. *Neural Processing Letters*, 54(5):4409–4426, 2022.
- [CUH16] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). In *ICLR*, 2016.
- [DBB⁺00] Charles Dugas, Yoshua Bengio, François Bélisle, Claude Nadeau, and René Garcia. Incorporating second-order functional knowledge for better option pricing. *Advances in neural information processing systems*, 13, 2000.
- [DBB⁺09] Charles Dugas, Yoshua Bengio, François Bélisle, Claude Nadeau, and René Garcia. Incorporating functional knowledge in neural networks. *Journal of Machine Learning Research*, 10(6), 2009.
- [DCD] Mengnan Du, Lu Cheng, and Dejing Dou. Algorithmic fairness in machine learning.
- [DEHH21] Christophe Denis, Romuald Elie, Mohamed Hebiri, and François Hu. Fairness guarantee in multi-class classification. *arXiv preprint arXiv:2109.13642*, 2021.
- [DG17a] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.
- [DG17b] Dheeru Dua and Casey Graff. UCI machine learning repository. <https://archive.ics.uci.edu/ml>, 2017. Accessed: 2025-05-12.

- [DHMS21] Frances Ding, Moritz Hardt, John Miller, and Ludwig Schmidt. Retiring adult: New datasets for fair machine learning. *Advances in neural information processing systems*, 34:6478–6490, 2021.
- [DHP⁺12] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226, 2012.
- [Die99] Thomas G. Dietterich. Hierarchical reinforcement learning with the maxq value function decomposition. *ArXiv*, cs.LG/9905014, 1999.
- [Din13] Steven X Ding. Fault diagnosis in feedback control systems and fault-tolerant architecture. *Model-Based Fault Diagnosis Techniques: Design Schemes, Algorithms and Tools*, pages 471–489, 2013.
- [DK25] Gregory Davrazos and Sotiris Kotsiantis. Evaluating fairness strategies in educational data mining: A comparative study of bias mitigation techniques. *Electronics*, 14, 05 2025.
- [DMC⁺23] David Dandolo, Chiara Masiero, Mattia Carletti, Davide Dalle Pezze, and Gian Antonio Susto. Acmeaccelerated model-agnostic explanations: Fast whitening of the machine-learning black box. *Expert Systems with Applications*, 214:119115, 2023.
- [DSC22] Shiv Ram Dubey, Satish Kumar Singh, and Bidyut Baran Chaudhuri. Activation functions in deep learning: A comprehensive survey and benchmark. *Neurocomputing*, 503:92–108, 2022.
- [DSM09] Gregory Druck, Burr Settles, and Andrew McCallum. Active learning by labeling features. In *Proceedings of the 2009 conference on Empirical methods in natural language processing*, pages 81–90, 2009.
- [DV10] Hennie Daniels and Marina Velikova. Monotone and partially monotone neural networks. *IEEE Transactions on Neural Networks*, 21(6):906–917, 2010.

- [DWD⁺16] Shubhomoy Das, Weng-Keen Wong, Thomas Dietterich, Alan Fern, and Andrew Emmott. Incorporating expert feedback into active anomaly discovery. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 853–858. IEEE, 2016.
- [DWF⁺17] Shubhomoy Das, Weng-Keen Wong, Alan Fern, Thomas G Dietterich, and Md Amran Siddiqui. Incorporating feedback into tree-based anomaly detection. *arXiv preprint arXiv:1708.09441*, 2017.
- [DYZH20] Mengnan Du, Fan Yang, Na Zou, and Xia Hu. Fairness in deep learning: A computational perspective. *IEEE Intelligent Systems*, 36(4):25–34, 2020.
- [EC] European Parliament and Council of the European Union. Regulation (EU) 2016/679 of the European Parliament and of the Council.
- [Edw98] M Knorr Edwin. Algorithms for mining distance-based outliers in large datasets. In *Proceedings of 24th international conference on very large databases (VLDB’98)*, pages 392–403, 1998.
- [EG18] Yanai Elazar and Yoav Goldberg. Adversarial removal of demographic attributes from text data. *arXiv preprint arXiv:1808.06640*, 2018.
- [ES15] Harrison Edwards and Amos Storkey. Censoring representations with an adversary. *arXiv preprint arXiv:1511.05897*, 2015.
- [FBD⁺24] Alessandro Fabris, Nina Baranowska, Matthew J Dennis, David Graus, Philipp Hacker, Jorge Saldivar, Frederik Zuiderveen Borgesius, and Asia J Biega. Fairness and bias in algorithmic hiring. *ACM Transactions on Intelligent Systems and Technology*, 2024.
- [FGD⁺21] Tharindu Fernando, Harshala Gammulle, Simon Denman, Sridha Sridharan, and Clinton Fookes. Deep learning for medical anomaly detection—a survey. *ACM Computing Surveys (CSUR)*, 54(7):1–37, 2021.

- [FK17] Mehdi Forouzanfar and Mohammad Javad Khosrowjerdi. A constrained optimization approach to integrated active fault detection and control. *Iranian Journal of Science and Technology, Transactions of Electrical Engineering*, 41:229–240, 2017.
- [FMSS22] Alessandro Fabris, Stefano Messina, Gianmaria Silvello, and Gian Antonio Susto. Algorithmic fairness datasets: the story so far. *Data Mining and Knowledge Discovery*, 36(6):2074–2152, 2022.
- [Foo21] Ralph Foorthuis. On the nature and types of anomalies: A review of deviations in data. *International Journal of Data Science and Analytics*, 12(4):297–331, 2021.
- [FYL⁺19] Rui Feng, Yang Yang, Yuehan Lyu, Chenhao Tan, Yizhou Sun, and Chunping Wang. Learning fair representations via an adversarial framework. *arXiv preprint arXiv:1904.13341*, 2019.
- [GB10] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.
- [GCP⁺16] Maya Gupta, Andrew Cotter, Jan Pfeifer, Konstantin Voevodski, Kevin Canini, Alexander Mangylov, Wojciech Moczydlowski, and Alexander Van Esbroeck. Monotonic calibrated interpolated look-up tables. *Journal of Machine Learning Research*, 17(109):1–47, 2016.
- [GFDVS21] Umang Gupta, Aaron M Ferber, Bistra Dilkina, and Greg Ver Steeg. Controllable guarantees for fair outcomes via contrastive information estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7610–7619, 2021.
- [GJDOP⁺21] David Gonzalez-Jimenez, Jon Del-Olmo, Javier Poza, Fernando Garramiola, and Patxi Madina. Data-driven fault diagnosis for electric drives: A review. *Sensors*, 21(12):4024, 2021.

- [GKP19] Paul Gözl, Anson Kahng, and Ariel D Procaccia. Paradoxes in fair machine learning. *Advances in Neural Information Processing Systems*, 32, 2019.
- [GMU24] Talia B. Gillis, Vitaly Meursault, and Berk Ustun. Operationalizing the search for less discriminatory alternatives in fair lending. In *The 2024 ACM Conference on Fairness, Accountability, and Transparency, FAccT 2024, Rio de Janeiro, Brazil, June 3-6, 2024*, pages 377–387. ACM, 2024.
- [GP18] Pratik Gajane and Mykola Pechenizkiy. On formalizing fairness in prediction with machine learning, 2018.
- [GPO08] Amol Ghoting, Srinivasan Parthasarathy, and Matthew Eric Otey. Fast mining of distance-based outliers in high-dimensional datasets. *Data Mining and Knowledge Discovery*, 16:349–364, 2008.
- [GSM⁺19] Akhil Gupta, Naman Shukla, Lavanya Marla, Arinbjörn Kolbeinsson, and Kartik Yellepeddi. How to incorporate monotonicity in deep networks while preserving flexibility? in *NeurIPS*, 2019.
- [GWFM⁺13] Ian Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. Maxout networks. In *International conference on machine learning*, pages 1319–1327. PMLR, 2013.
- [GY19] Seyed Hossein Ghafarian and Hadi Sadoghi Yazdi. Functional gradient approach to probabilistic minimax active learning. *Engineering Applications of Artificial Intelligence*, 85:21–32, 2019.
- [H⁺02] Bernhard Hengst et al. Discovering hierarchy in reinforcement learning with hexq. In *Icml*, volume 19, pages 243–250. Cite-seer, 2002.
- [HAB19] Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf. Why relu networks yield high-confidence predictions far

- away from the training data and how to mitigate the problem. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 41–50, 2019.
- [HAC⁺21] Mark K. Ho, David Abel, Carlos G Correa, Michael L. Littman, Jonathan D. Cohen, and Thomas L. Griffiths. People construct simplified mental representations to plan. *Nature*, 606:129 – 136, 2021.
- [HAGL19] Mark K. Ho, David Abel, Thomas L. Griffiths, and Michael L. Littman. The value of abstraction. *Current Opinion in Behavioral Sciences*, 29:111–116, 2019.
- [Haw80] Douglas M Hawkins. *Identification of outliers*, volume 11. Springer, 1980.
- [HHGL11] Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *ArXiv*, abs/1112.5745, 2011.
- [HHH⁺22a] Songqiao Han, Xiyang Hu, Hailiang Huang, Minqi Jiang, and Yue Zhao. Adbench: Anomaly detection benchmark. *Advances in Neural Information Processing Systems*, 35:32142–32159, 2022.
- [HHH⁺22b] Songqiao Han, Xiyang Hu, Hailiang Huang, Minqi Jiang, and Yue Zhao. Adbench: Anomaly detection benchmark. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 32142–32159. Curran Associates, Inc., 2022.
- [HKB19] Sahand Hariri, Matias Carrasco Kind, and Robert J Brunner. Extended isolation forest. *IEEE transactions on knowledge and data engineering*, 33(4):1479–1489, 2019.
- [HM10] David J Hill and Barbara S Minsker. Anomaly detection in streaming environmental sensor data: A data-driven modeling approach. *Environmental Modelling & Software*, 25(9):1014–1022, 2010.

- [HM17] Tor Aksel N. Heirung and Ali Mesbah. Stochastic nonlinear model predictive control with active model discrimination: A closed-loop fault diagnosis application. *IFAC-PapersOnLine*, 2017.
- [Ho95] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.
- [HPS16] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29, 2016.
- [HS18] Julia Hofmockel and Eric Sax. Isolation forest for anomaly detection in raw vehicle sensor data. In *VEHITS*, pages 411–416, 2018.
- [HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [Iof15] Sergey Ioffe. Batch normalization: Accelerating deep network training by reducing internal covariate shift. in *ICML*, 2015.
- [JA08] Sheng-yi Jiang and Qing-bo An. Clustering-based outlier detection method. In *2008 Fifth international conference on fuzzy systems and knowledge discovery*, volume 2, pages 429–433. IEEE, 2008.
- [JGLM23] Danial Javaheri, Saeid Gorgin, Jeong-A Lee, and Mohammad Masdari. Fuzzy logic-based ddos attacks and network traffic anomaly detection methods: Classification, overview, and future perspectives. *Information Sciences*, 626:315–338, 2023.
- [JH21] Fanlin Jia and Xiao He. Auxiliary input design for active fault detection via deep reinforcement learning. In *CAA SAFEPROCESS*, 2021.
- [JMVS⁺20] Ayush Jaiswal, Daniel Moyer, Greg Ver Steeg, Wael AbdAlmageed, and Premkumar Natarajan. Invariant representations

- through adversarial forgetting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4272–4279, 2020.
- [JPS⁺20] Ray Jiang, Aldo Pacchiano, Tom Stepleton, Heinrich Jiang, and Silvia Chiappa. Wasserstein fair classification. In *Uncertainty in artificial intelligence*, pages 862–872. PMLR, 2020.
- [JTS01] Mon-Fong Jiang, Shian-Shyong Tseng, and Chih-Ming Su. Two-phase clustering process for outliers detection. *Pattern recognition letters*, 22(6-7):691–700, 2001.
- [JXF⁺16] Xiaojie Jin, Chunyan Xu, Jiashi Feng, Yunchao Wei, Junjun Xiong, and Shuicheng Yan. Deep learning with s-shaped rectified linear activation units. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.
- [JYTB23] Vincent Jeanselme, Chang Ho Yoon, Brian Tom, and Jessica Barrett. Neural fine-gray: Monotonic neural networks for competing risks. In *Conference on Health, Inference, and Learning*, pages 379–392. PMLR, 2023.
- [KB09] George Dimitri Konidaris and Andrew G. Barto. Efficient skill learning using abstraction selection. In *International Joint Conference on Artificial Intelligence*, 2009.
- [KB14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [KEE⁺15] Konstantina Kourou, Themis P Exarchos, Konstantinos P Exarchos, Michalis V Karamouzis, and Dimitrios I Fotiadis. Machine learning applications in cancer prognosis and prediction. *Computational and structural biotechnology journal*, 13:8–17, 2015.
- [KG20] Punit Kumar and Atul Gupta. Active learning query strategies for classification, regression, and clustering: a survey. *Journal of Computer Science and Technology*, 35(4):913–945, 2020.
- [KKLP15] George Dimitri Konidaris, Leslie Pack Kaelbling, and Tomas Lozano-Perez. Symbol acquisition for probabilistic high-level

- planning. In *International Joint Conference on Artificial Intelligence*, 2015.
- [KKSZ09] Hans-Peter Kriegel, Peer Kröger, Erich Schubert, and Arthur Zimek. Loop: local outlier probabilities. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 1649–1652, 2009.
- [KL24] Hyunho Kim and Jong-Seok Lee. Scalable monotonic neural networks. In *The Twelfth International Conference on Learning Representations*, 2024.
- [KN98] Edwin M Knorr and Raymond T Ng. Algorithms for mining distance-based outliers in large datasets. In *VLDB*, volume 98, pages 392–403. Citeseer, 1998.
- [KP22] Ladislav Krl and Ivo Punoch. Policy search for active fault diagnosis with partially observable state. *International Journal of Adaptive Control and Signal Processing*, 36(9):2190–2216, 2022.
- [KSZ08] Hans-Peter Kriegel, Matthias Schubert, and Arthur Zimek. Angle-based outlier detection in high-dimensional data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 444–452, 2008.
- [KUMH17] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. *Advances in neural information processing systems*, 30, 2017.
- [KvAG19] Andreas Kirsch, Joost R. van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. *ArXiv*, abs/1906.08158, 2019.
- [LAR⁺19] Francesco Locatello, Gabriele Abbati, Thomas Rainforth, Stefan Bauer, Bernhard Schölkopf, and Olivier Bachem. On the fairness of disentangled representations. *Advances in neural information processing systems*, 32, 2019.

- [LBST21a] Julien Lesouple, Cédric Baudoin, Marc Spigai, and Jean-Yves Tournieret. Generalized isolation forest for anomaly detection. *Pattern Recognition Letters*, 149:109–119, 2021.
- [LBST21b] Julien Lesouple, Cédric Baudoin, Marc Spigai, and Jean-Yves Tournieret. How to introduce expert feedback in one-class support vector machines for anomaly detection? *Signal Processing*, 188:108197, 2021.
- [LG04] Teresa Lynch and Shirley Gregor. User participation in decision support systems development: influencing system outcomes. *European Journal of Information Systems*, 13:286–301, 2004.
- [LHZL20] Xingchao Liu, Xing Han, Na Zhang, and Qiang Liu. Certified monotonic neural networks. *Advances in Neural Information Processing Systems*, 33:15427–15438, 2020.
- [LRT14] Balaji Lakshminarayanan, Daniel M Roy, and Yee Whye Teh. Mondrian forests: Efficient online random forests. *Advances in neural information processing systems*, 27, 2014.
- [LSL⁺17] Christos Louizos, Kevin Swersky, Yujia Li, Max Welling, and Richard Zemel. The variational fair autoencoder, 2017.
- [LTZ08] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. *2008 eighth ieee international conference on data mining*, pages 413–422, ”2008.
- [LTZ09] Fei Tony Liu, Kai Ting, and Zhi-Hua Zhou. Isolation forest. In *Proceedings of the IEEE International Conference on Data Mining*, 01 2009.
- [LTZ12] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6(1):1–39, 2012.
- [Luk70] Eugene Lukacs. Characteristic functions. *(No Title)*, 1970.

- [LYQ⁺23] Yang Liu, Guoping Yang, Shaojie Qiao, Meiqi Liu, Lulu Qu, Nan Han, Tao Wu, Guan Yuan, and Yuzhong Peng. Imbalanced data classification: Using transfer learning and active sampling. *Engineering Applications of Artificial Intelligence*, 117:105621, 2023.
- [LB⁺23] Daniel Leite, Igor krjanc, Sao Blai, Andrej Zdear, and Fernando Gomide. Interval incremental learning of interval data streams and application to vehicle tracking. *Information Sciences*, 630:1–22, 2023.
- [MBS22] Elisa Marcelli, Tommaso Barbariol, and Gian Antonio Susto. Active learning-based isolation forest (alif): Enhancing anomaly detection in decision support systems. *arXiv preprint arXiv:2207.03934*, 2022.
- [MBSS24] Elisa Marcelli, Tommaso Barbariol, Davide Sartor, and Gian Antonio Susto. Active learning-based isolation forest (alif): Enhancing anomaly detection with expert feedback. *Information Sciences*, 678:121012, 2024.
- [MCPZ18] David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. Learning adversarially fair and transferable representations. In *International Conference on Machine Learning*, pages 3384–3393. PMLR, 2018.
- [MFCC⁺16] Mahdi Milani Fard, Kevin Canini, Andrew Cotter, Jan Pfeifer, and Maya Gupta. Fast and flexible monotonic functions with ensembles of lattices. *Advances in neural information processing systems*, 29, 2016.
- [MGB⁺18] Daniel Moyer, Shuyang Gao, Rob Brekelmans, Aram Galstyan, and Greg Ver Steeg. Invariant representations without adversarial training. *Advances in neural information processing systems*, 31, 2018.
- [MGS⁺20] Haoran Ma, Benyamin Ghogh, Maria N Samad, Dongyu Zheng, and Mark Crowley. Isolation mondrian forest for batch and online anomaly detection. In *2020 IEEE International*

- Conference on Systems, Man, and Cybernetics (SMC)*, pages 3051–3058. IEEE, 2020.
- [Mit80] Tom M Mitchell. The need for biases in learning generalizations. Technical Report CBM-TR-117, Carnegie Mellon University, 1980.
- [MKMM20] Dariusz Mrozek, Anna Koczur, and Boena Maysiak-Mrozek. Fall detection in older adults with mobile iot devices and machine learning in the cloud and on the edge. *Information Sciences*, 537:132–147, 2020.
- [MNCHH20] Norazwan Md Nor, Che Rosmani Che Hassan, and Mohd Azlan Hussain. A review of data-driven fault detection and diagnosis methods: Applications in chemical process systems. *Reviews in Chemical Engineering*, 36(4):513–553, 2020.
- [MOW19] Daniel McNamara, Cheng Soon Ong, and Robert C. Williamson. Costs and benefits of fair representation learning. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, AIES ’19, page 263270, New York, NY, USA, 2019. Association for Computing Machinery.
- [MR17] G Roberto Marseglia and Davide M Raimondo. Active fault diagnosis: A multi-parametric approach. *Automatica*, 79:223–230, 2017.
- [MR22] Dan Mikulincer and Daniel Reichman. Size and depth of monotone neural networks: interpolation and approximation. *Advances in Neural Information Processing Systems*, 35:5522–5534, 2022.
- [MW20] Ramanujam Madhavan and Mohit Wadhwa. Fairness-aware learning with prejudice free representations. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM ’20, page 21372140, New York, NY, USA, 2020. Association for Computing Machinery.
- [NH10] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the*

- 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.
- [NJG17] Gergely Neu, Anders Jonsson, and V. Gómez. A unified view of entropy-regularized markov decision processes. *ArXiv*, 2017.
 - [NKW23] Niklas Nolte, Ouail Kitouni, and Mike Williams. Expressive monotonic neural networks. In *The Eleventh International Conference on Learning Representations*, 2023.
 - [NM19] An-phi Nguyen and María Rodríguez Martínez. Mononet: towards interpretable models by learning monotonic features. In *NeurIPS*, 2019.
 - [NTND21] Ali Bou Nassif, Manar Abu Talib, Qassim Nasir, and Fatima Mohamad Dakalbab. Machine learning for anomaly detection: A systematic review. *Ieee Access*, 9:78658–78700, 2021.
 - [OA⁺19] Takahiro Omi, Kazuyuki Aihara, et al. Fully neural network based model for general temporal point processes. *Advances in neural information processing systems*, 32, 2019.
 - [OPVM19] Ziad Obermeyer, Brian Powers, Christine Vogeli, and Sendhil Mullainathan. Dissecting racial bias in an algorithm used to manage the health of populations. *Science*, 366(6464):447–453, Oct 2019.
 - [PC97] Ron J Patton and Jie Chen. Observer-based fault detection and isolation: Robustness and applications. *Control Engineering Practice*, 5(5):671–682, 1997.
 - [Pev16] Tomáš Pevný. Loda: Lightweight on-line detector of anomalies. *Machine Learning*, 102:275–304, 2016.
 - [PGM⁺19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

- [PHBM18] Joel A. Paulson, Tor Aksel N. Heirung, Richard D. Braatz, and Ali Mesbah. Closed-loop active fault diagnosis for stochastic linear systems. In *2018 Annual American Control Conference (ACC)*, 2018.
- [PKGf03] Spiros Papadimitriou, Hiroyuki Kitagawa, Phillip B Gibbons, and Christos Faloutsos. Loci: Fast outlier detection using the local correlation integral. In *Proceedings 19th international conference on data engineering (Cat. No. 03CH37405)*, pages 315–326. IEEE, 2003.
- [PM04] Dan Pelleg and Andrew W. Moore. Active learning for anomaly and rare-category detection. In *Neural Information Processing Systems*, 2004.
- [PMCM17] Joel Paulson, Marc Martin-Casas, and Ali Mesbah. Input design for online fault diagnosis of nonlinear systems with stochastic uncertainty. *Industrial & Engineering Chemistry Research*, 2017.
- [PSJvdH20] Guansong Pang, Chunhua Shen, Huidong Jin, and Anton van den Hengel. Deep weakly-supervised anomaly detection, 2020.
- [PSK22] Evaggelia Pitoura, Kostas Stefanidis, and Georgia Koutrika. Fairness in rankings and recommendations: an overview. *The VLDB Journal*, pages 1–28, 2022.
- [P15] Ivo Punoch, Jan irok, and Miroslav imandl. Constrained active fault detection and control. *IEEE Trans. on Automatic Control*, 2015.
- [Ray16] Shebuti Rayana. ODDS library, 2016.
- [RB03] Balaraman Ravindran and Andrew G. Barto. Smdp homomorphisms: An algebraic approach to abstraction in semi-markov decision processes. In *International Joint Conference on Artificial Intelligence*, 2003.
- [RB19] Proteek Chandan Roy and Vishnu Naresh Boddeti. Mitigating information leakage in image representations: A maximum

- entropy approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2586–2594, 2019.
- [RLWS20] Yuji Roh, Kangwook Lee, Steven Whang, and Changho Suh. Fr-train: A mutual information-based approach to fair and robust training. In *International Conference on Machine Learning*, pages 8147–8157. PMLR, 2020.
- [RN18] Jacob Rafati and David C. Noelle. Learning representations in model-free hierarchical reinforcement learning. *ArXiv*, abs/1810.10096, 2018.
- [RPK⁺17] Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein. On the expressive power of deep neural networks. In *international conference on machine learning*, pages 2847–2854. PMLR, 2017.
- [RS23a] Aswathy Ravikumar and Harini Sriraman. Mitigating vanishing gradient in sgd optimization in neural networks. In *International Conference on Information, Communication and Computing Technology*, pages 1–11. Springer, 2023.
- [RS23b] Davor Runje and Sharath M Shankaranarayana. Constrained monotonic neural networks. In *International Conference on Machine Learning*, pages 29338–29353. PMLR, 2023.
- [RVG⁺19] Lukas Ruff, Robert A Vandermeulen, Nico Görnitz, Alexander Binder, Emmanuel Müller, Klaus-Robert Müller, and Marius Kloft. Deep semi-supervised anomaly detection. *arXiv preprint arXiv:1906.02694*, 2019.
- [RXC⁺21] Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B Gupta, Xiaojiang Chen, and Xin Wang. A survey of deep active learning. *ACM computing surveys (CSUR)*, 54(9):1–40, 2021.
- [R21] Tim Rz. Group fairness: Independence revisited. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT 21, page 129137. ACM, March 2021.

- [Sac74] Earl D. Sacerdoti. Planning in a hierarchy of abstraction spaces. *Artif. Intell.*, 5:115–135, 1974.
- [SAM96] Joseph Sill and Yaser Abu-Mostafa. Monotonicity hints. *Advances in neural information processing systems*, 9, 1996.
- [SBM17] Gian Antonio Susto, Alessandro Beghi, and Seán McLoone. Anomaly detection through on-line isolation forest: An application to plasma etching. In *2017 28th Annual SEMI Advanced Semiconductor Manufacturing Conference (ASMC)*, pages 89–94. IEEE, 2017.
- [SBS24a] Davide Sartor, Tommaso Barbariol, and Gian Antonio Susto. Bayesian active learning isolation forest (b-alif): A weakly supervised strategy for anomaly detection. *Engineering Applications of Artificial Intelligence*, 130:107671, 2024.
- [SBS24b] Davide Sartor, Tommaso Barbariol, and Gian Antonio Susto. Bayesian active learning isolation forest (b-alif): A weakly supervised strategy for anomaly detection. *Engineering Applications of Artificial Intelligence*, 130:107671, 2024.
- [SC12] David Silver and Kamil Ciosek. Compositional planning using optimal option models. In *ICML*, 2012.
- [SCC23] Luciano Sánchez, Nahuel Costa, and Inés Couso. Simplified models of remaining useful life based on stochastic orderings. *Reliability Engineering & System Safety*, 237:109321, 2023.
- [Set95] Burr Settles. Active learning literature survey. *Science*, 10(3):237–304, 1995.
- [Set11] Burr Settles. From theories to queries: Active learning in practice. In *Active learning and experimental design workshop in conjunction with AISTATS 2010*, pages 1–18. JMLR Workshop and Conference Proceedings, 2011.
- [SFMVdB20] Aishwarya Sivaraman, Golnoosh Farnadi, Todd Millstein, and Guy Van den Broeck. Counterexample-guided learning of monotonic neural networks. *Advances in Neural Information Processing Systems*, 33:11936–11948, 2020.

- [SGC17] Swati Shilaskar, Ashok Ghatol, and Prashant Chatur. Medical decision support system for extremely imbalanced datasets. *Information Sciences*, 384:205–219, 2017.
- [She16a] Rayana Shebuti. Odds library, 2016.
- [She16b] Rayana Shebuty. ODDS library, 2016.
- [SHG⁺23] Sunzida Siddique, Mohd Ariful Haque, Roy George, Kishor Datta Gupta, Debashis Gupta, and Md Jobair Hos-sain Faruk. Survey on machine learning biases and mitigation techniques. *Digital*, 4(1):1–68, 2023.
- [SKG⁺19] Jiaming Song, Pratyusha Kalluri, Aditya Grover, Shengjia Zhao, and Stefano Ermon. Learning controllable fair representations. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2164–2173. PMLR, 2019.
- [SKHN21] Edin Sabic, David Keeley, Bailey Henderson, and Sara Nannemann. Healthcare and anomaly detection: using machine learning to predict anomalies in heart rate data. *AI & SOCIETY*, 36, 03 2021.
- [SKLM⁺20] Laleh Seyyed-Kalantari, Guanxiong Liu, Matthew McDermott, Irene Y. Chen, and Marzyeh Ghassemi. Chexclusion: Fairness gaps in deep chest x-ray classifiers, 2020.
- [SPS99] Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.
- [SPY⁺24] Hyorim Shin, Jeongeun Park, Jeongmin Yu, Jungeun Kim, Ha Young Kim, and Changhoon Oh and. Looping in: Exploring feedback strategies to motivate human engagement in interactive machine learning. *International Journal of Human-Computer Interaction*, 0(0):1–18, 2024.
- [SR15] Takaya Saito and Marc Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary

- classifiers on imbalanced datasets. *PloS one*, 10(3):e0118432, 2015.
- [SS19] Congzheng Song and Vitaly Shmatikov. Overlearning reveals sensitive attributes. *arXiv preprint arXiv:1905.11742*, 2019.
- [SSK21] Jonas Herskind Sejr and Anna Schneider-Kamp. Explainable outlier detection: What, for whom and why? *Machine Learning with Applications*, 6:100172, 2021.
- [SWS⁺99] Bernhard Schölkopf, Robert C Williamson, Alexander J Smola, John Shawe-Taylor, John C Platt, et al. Support vector method for novelty detection. In *NIPS*, volume 12, pages 582–588. Citeseer, 1999.
- [Sza21] Tomasz Szandala. Review and comparison of commonly used activation functions for deep neural networks. *Bio-inspired neurocomputing*, pages 203–224, 2021.
- [TCFC02] Jian Tang, Zhixiang Chen, Ada Wai-Chee Fu, and David W Cheung. Enhancing effectiveness of outlier detections for low density patterns. In *Advances in Knowledge Discovery and Data Mining: 6th Pacific-Asia Conference, PAKDD 2002 Taipei, Taiwan, May 6–8, 2002 Proceedings 6*, pages 535–548. Springer, 2002.
- [TLP19] Natasa Tagasovska and David Lopez-Paz. Single-model uncertainties for deep learning. *Advances in neural information processing systems*, 32, 2019.
- [TOSX21] Junbo Tan, Sorin Olaru, Maria M Seron, and Feng Xu. Set-based guaranteed active fault diagnosis for lpv systems with unknown bounded uncertainties. *Automatica*, 128:109602, 2021.
- [TS22] Mycal Tucker and Julie A Shah. Prototype based classification from hierarchy to fairness. In *International Conference on Machine Learning*, pages 21884–21900. PMLR, 2022.

- [TXWL22] Junbo Tan, Feng Xu, Yushuai Wang, and Bin Liang. Input design for integrated active fault diagnosis and output tracking control. *Automatica*, 2022.
- [Vas17] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.
- [Vel19] Petar Veličković. *The resurgence of structure in deep neural networks*. PhD thesis, University of Cambridge, 2019.
- [VTM21] Antonio Vetrò, Marco Torchiano, and Mariachiara Mecati. A data quality approach to the identification of discrimination risk in automated decision making systems. *Gov. Inf. Q.*, 38(4):101619, 2021.
- [VZH⁺21] Vivek Veeriah, Tom Zahavy, Matteo Hessel, Zhongwen Xu, Junhyuk Oh, Iurii Kemaev, H. V. Hasselt, David Silver, and Satinder Singh. Discovery of options via meta-learned sub-goals. In *Neural Information Processing Systems*, 2021.
- [WL19] Antoine Wehenkel and Gilles Louppe. Unconstrained monotonic neural networks. *Advances in neural information processing systems*, 32, 2019.
- [WO21] Tong Wu and Jorge Ortiz. Rlad: Time series anomaly detection through reinforcement learning and active learning. *arXiv preprint arXiv:2104.00543*, 2021.
- [Woo50] Max A Woodbury. *Inverting modified matrices*. Department of Statistics, Princeton University, 1950.
- [WWL⁺23] Lin Wang, Xiangjun Wang, Feng Liu, Mingyang Li, Xin Hao, and Nianfu Zhao. Attention-guided mil weakly supervised visual anomaly detection. *Measurement*, page 112500, 2023.
- [WZW18] Yongkai Wu, Lu Zhang, and Xintao Wu. Fairness-aware classification: Criterion, convexity, and bounds. *arXiv preprint arXiv:1809.04737*, 2018.
- [WZZ22] Xiaomeng Wang, Yishi Zhang, and Ruilin Zhu. A brief review on algorithmic fairness. *Management System Engineering*, 1(1):7, 2022.

- [XDD⁺17] Qizhe Xie, Zihang Dai, Yulun Du, Eduard Hovy, and Graham Neubig. Controllable invariance through adversarial feature learning. *Advances in neural information processing systems*, 30, 2017.
- [XOP⁺17] Feng Xu, Sorin Olaru, Vicenc Puig, Carlos Ocampo-Martinez, and Silviu-Iulian Niculescu. Sensor-fault tolerance using robust mpc with set-based state estimation and active fault isolation. *International Journal of Robust and Nonlinear Control*, 27(8):1260–1283, 2017.
- [Xu21] Feng Xu. Observer-based asymptotic active fault diagnosis: A two-layer optimization framework. *Automatica*, 128, 2021.
- [XZS⁺20] Yilun Xu, Shengjia Zhao, Jiaming Song, Russell Stewart, and Stefano Ermon. A theory of usable information under computational constraints. *arXiv preprint arXiv:2002.10689*, 2020.
- [YCM24] Zhe Yu, Joymallya Chakraborty, and Tim Menzies. Fairbalance: How to achieve equalized odds with data pre-processing. *IEEE Transactions on Software Engineering*, 50(9):22942312, September 2024.
- [YDC⁺17] Seungil You, David Ding, Kevin Canini, Jan Pfeifer, and Maya Gupta. Deep lattice networks and partial monotonic functions. *Advances in neural information processing systems*, 30, 2017.
- [YLL⁺22] Zhen Yang, Xiaodong Liu, Tong Li, Di Wu, Jinjiang Wang, Yunwei Zhao, and Han Han. A systematic literature review of methods and datasets for anomaly-based network intrusion detection. *Computers & Security*, 116:102675, 2022.
- [YMK22] Hiroki Yanagisawa, Kohei Miyaguchi, and Takayuki Katsuki. Hierarchical lattice layer for partially monotone neural networks. *Advances in Neural Information Processing Systems*, 35:11092–11103, 2022.
- [YXT⁺23] Zichen Yan, Feng Xu, Junbo Tan, Houde Liu, and Bin Liang. Reinforcement learning-based integrated active fault diagnosis and tracking control. *ISA transactions*, 132:364–376, 2023.

- [ZG09] Xiaojin Zhu and Andrew B Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130, 2009.
- [ZH20] Zhao Zhang and Xiao He. Fault-structure-based active fault diagnosis: A geometric observer approach. *Energies*, 13(17):4475, 2020.
- [ZHBN19] Patrick Zschech, Kai Heinrich, Raphael Bink, and Janis S Neufeld. Prognostic model development with missing labels: A condition-based maintenance approach using machine learning. *Business & Information Systems Engineering*, 61:327–343, 2019.
- [ZHHL20] Yong-Ping Zhao, Gong Huang, Qian-Kun Hu, and Bing Li. An improved weighted one class support vector machine for turboshaft engine fault detection. *Engineering Applications of Artificial Intelligence*, 94:103796, 2020.
- [ZLQ⁺22] Chen Zhang, Guorong Li, Yuankai Qi, Shuhui Wang, Laiyun Qing, Qingming Huang, and Ming-Hsuan Yang. Exploiting completeness and uncertainty of pseudo labels for weakly supervised video anomaly detection. *arXiv preprint arXiv:2212.04090*, 2022.
- [ZLQ⁺23] Chen Zhang, Guorong Li, Yuankai Qi, Shuhui Wang, Laiyun Qing, Qingming Huang, and Ming-Hsuan Yang. Exploiting completeness and uncertainty of pseudo labels for weakly supervised video anomaly detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16271–16280, 2023.
- [ZT24] Meiyu Zhong and Ravi Tandon. Intrinsic fairness-accuracy tradeoffs under equalized odds. In *2024 IEEE International Symposium on Information Theory (ISIT)*, pages 220–225. IEEE, 2024.
- [ZVF⁺23] Justus Zipfel, Felix Verworner, Marco Fischer, Uwe Wieland, Mathias Kraus, and Patrick Zschech. Anomaly detection for

industrial quality assurance: A comparative evaluation of unsupervised deep learning models. *Computers & Industrial Engineering*, 177:109045, 2023.

- [ZVRG17] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P. Gummadi. Fairness Constraints: Mechanisms for Fair Classification. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 962–970. PMLR, 20–22 Apr 2017.
- [ZWS⁺13] Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. Learning fair representations. In *International conference on machine learning*, pages 325–333. PMLR, 2013.
- [PL16] Jan kach, Ivo Punoch, and Frank L. Lewis. Optimal active fault diagnosis by temporal-difference learning. In *IEEE CDC*, 2016.