ProtoHedge: Interpretable Hedging with Market Prototypes

Lisa Faloughi
Imperial College London
United Kingdom
lisa.faloughi24@imperial.ac.uk

Ce Guo
Imperial College London
United Kingdom
c.guo@imperial.ac.uk

Wayne Luk
Imperial College London
United Kingdom
w.luk@imperial.ac.uk

Abstract

Deep hedging has emerged as a powerful framework for financial risk management, capable of learning effective hedging strategies in the presence of market frictions. However, its reliance on black-box neural networks creates a critical barrier to adoption, limiting trust, auditability, and regulatory compliance. In this work, we address this challenge by introducing a transparent alternative to the black-box Deep Hedging paradigm. Instead of relying on an opaque architecture, our model learns a finite set of representative market states, or "prototypes". Hedging decisions are then made via a transparent, similarity-based mechanism: the agent's action is a weighted average of learned actions associated with each prototype. We call our specific implementation of this framework ProtoHedge. The reasoning approach makes every decision traceable to understandable market scenarios. We conduct extensive experiments in both classical Black-Scholes and more realistic stochastic volatility environments. Our results demonstrate that this interpretability is achieved with minimal impact of less than 0.40% on hedging performance, as our model's hedging effectiveness is comparable to that of the original black-box deep hedging agent. This work shows that transparency and performance are not mutually exclusive, paving the way for more trustworthy automated risk management systems.

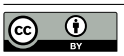
CCS Concepts

• **Computing methodologies** → **Instance-based learning**; **Sequential decision making**; *Online learning settings*; *Neural networks*; • **Applied computing** → **Economics**.

Keywords

deep hedging, explainable AI, utility-based reinforcement learning, intrinsic interpretability, risk management

ACM Reference Format:
Lisa Faloughi, Ce Guo, and Wayne Luk. 2025. ProtoHedge: Interpretable Hedging with Market Prototypes. In *6th ACM International Conference on AI in Finance (ICAIF '25), November 15–18, 2025, Singapore, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3768292.3770347



ICAIF '25, Singapore, Singapore

ACM ISBN 979-8-4007-2220-2/2025/11
https://doi.org/10.1145/3768292.3770347

1 Introduction

The Deep Hedging framework, introduced by Buehler et al. [8], represents a significant advance in computational finance. By reformulating derivatives hedging as a data-driven, sequential decision-making problem, it learns robust strategies that can manage real-world complexities like transaction costs, market frictions, and incomplete information—challenges that often confound classical, model-based approaches. The framework's ability to optimize for an investor's specific risk preferences (utility) further enhances its practical appeal.

However, Deep Hedging has a critical limitation: its black-box nature. The hedging policy is determined by a deep neural network whose internal logic is opaque. This lack of transparency poses a major obstacle for deployment in the financial industry, where trust, auditability, and regulatory compliance are critical [7]. As with other critical AI applications, there is a significant risk that the model could learn spurious, non-causal correlations from the data, leading to unpredictable failures in live market conditions [30].

In response to this challenge, the field of eXplainable AI (XAI) has broadly evolved into two paradigms. The first, *post-hoc explanation*, attempts to explain the decisions of a pre-trained black-box model. Methods like LIME [22] and SHAP [17] operate by building a secondary, simpler model to approximate the behavior of the complex model around a single prediction. However, a growing body of research has exposed their fundamental flaws, showing they can be unstable, misleading, and even manipulated [16, 26]. These limitations have motivated a shift towards the second paradigm: *intrinsic interpretability*, where models are transparent by design. In an intrinsically interpretable model, the structure itself is sufficiently understandable that the calculation leading to the output *is* the explanation. Classic examples include linear models, decision trees, and rule-based systems. While these models are highly transparent, they often lack the expressive capacity to capture the complex, non-linear dynamics inherent in financial markets.

This paper presents a novel architecture designed to retain the modeling power necessary for deep hedging while providing the structural transparency of case-based reasoning. Instead of relying on a complex, opaque network, our model learns a discrete set of representative market states (prototypes). At each time step, it computes the similarity between the current market conditions and these learned prototypes, and the final hedging action is a transparent, weighted combination of the learned actions tied to each prototype. We call our specific implementation of this framework ProtoHedge. This design ensures that every decision is directly attributable to a collection of understandable, historical scenarios. Crucially, we demonstrate that this gain in interpretability does not require a significant sacrifice in hedging effectiveness.

The practical implications of this interpretability are deep. For example, during a sudden market downturn, a conventional deep

hedging model can only justify its actions through its internal numerical states, such as a specific pattern of neuron activations, which provides little financial insight. In contrast, our proposed framework offers a causally transparent rationale. It would explain a large defensive trade by identifying high similarity to established prototypes, for example, a "2008-style financial crisis" or a "2020-style flash crash". Since the influence of each prototype and its corresponding learned action is explicit, the model's reasoning becomes directly auditable and aligns with case-based analysis.

Our main contributions are:

- A novel interpretable architecture and training methodology, where we present the complete ProtoHedge model, its two-phase training process, and the derivation of its analytical gradients for end-to-end optimization (Section 3).
- An empirical evaluation of hedging performance, demonstrating through extensive experiments that our transparent model achieves hedging effectiveness comparable to a black-box agent and successfully replicates the optimal analytical solution in a classical market (Section 4.2).
- A practical framework for model auditing and interpretation, where we demonstrate how the model's transparency can be used to trace any specific hedging decision back to understandable prototypes and to verify the global financial logic learned by the model (Sections 4.3 and 4.4).

The remainder of this paper details our proposed approach, presents the experimental results validating its performance and interpretability, and discusses its implications for building more trustworthy financial AI systems. The code for our implementation is available at https://github.com/sonnets-project/protohedge.

2 Background and Related Work

2.1 Black-Box Deep Hedging

The Deep Hedging framework [8] formulates hedging as a sequential decision-making problem. Over a discrete time grid $\mathcal{T} = \{t_0, \ldots, t_N = T\}$, an agent with a policy F_θ observes the market state s_{t_i} and outputs an action $a^\theta_{t_i} = F_\theta(s_{t_i})$. This action represents the *change* in the hedge position. The total position (or hedge), $\delta^\theta_{t_i} \in \mathbb{R}^d$, is the cumulative sum of actions, updated as $\delta^\theta_{t_i} = \delta^\theta_{t_{i-1}} + a^\theta_{t_i}$, with an initial position of $\delta^\theta_{t_{-1}} = \mathbf{0}$.

The agent is trained to manage the terminal Profit and Loss (P&L), which is a function of the policy parameters θ. For a portfolio started with an initial premium p_0, the P&L is:

$$\mathrm{PL}^\theta_T = p_0 - Z_T + (\delta^\theta \cdot \Delta S)_T - C^\theta_T \tag{1}$$

where Z_T is the contingent liability, $(\delta^\theta \cdot \Delta S)_T$ is the cumulative gain from the self-financing trading strategy, and C^θ_T represents the policy-dependent cumulative transaction costs, which are typically proportional to the value of each trade. The trading gain is calculated as:

$$(\delta^\theta \cdot \Delta S)_T = \sum_{i=0}^{N-1} \delta^\theta_{t_i} \cdot (S_{t_{i+1}} - S_{t_i}) \tag{2}$$

The training objective is to find the optimal policy parameters θ that maximize the hedger's risk preference, typically framed as maximizing the expected utility $U(\cdot)$ of the terminal P&L:

$$\max_\theta \mathbb{E}[U(\mathrm{PL}^\theta_T)] \tag{3}$$

This allows the framework to learn strategies that align with a specific risk measure, like Conditional Value-at-Risk (CVaR), without relying on classical assumptions of complete markets.

Recent extensions enhance the real-world applicability of the Deep Hedging framework by addressing key practical challenges, including the shift to continuous-time trading, the impact of transaction costs, and validation on empirical data. For instance, Murray et al. [19] adapt the framework for continuous-time settings using stochastic control, improving its accuracy for high-frequency instruments. To manage transaction costs, Imaki et al. [14] introduce a No-Transaction Band Network that prevents costly overtrading. Mikkilä and Kanniainen [18] validate the framework's potential in a fully model-free setting, showing that an agent trained directly on real S&P 500 option data can outperform classical and model-based methods. In a related line of work, Kolm and Ritter [15] study how transaction costs in discrete-time settings create systematic deviations from Black–Scholes delta hedging, highlighting empirical links between option moneyness and hedging actions. Vittori, Trapletti, and Restelli [29] extend this perspective using reinforcement learning in a risk-averse setting, explicitly analyzing how moneyness and trading costs jointly shape the learned policy, and showing that such deviations can outperform the classical Black–Scholes hedge.

A fundamental limitation of Deep Hedging is its black-box nature, which creates challenges for trust, auditing, and regulatory compliance. The creators of deep hedging acknowledge this, noting that the neural network's decision-making logic is opaque [8]. This opacity is a widely recognized problem in applied AI [7], creating ethical and operational risks [30] because models can learn spurious, non-causal correlations. This danger is well-documented in critical fields like medicine, where systems have mistakenly learned to associate chest tubes with pneumothorax [25] or surgical rulers with skin lesions [6] instead of the actual clinical features. Consequently, addressing this fundamental lack of transparency has become a central goal in modern AI, giving rise to the field of explainable AI.

2.2 Explainable AI for Finance

In response to the opacity in black-box models, a field of post-hoc explainability emerged, with model-agnostic methods like LIME [22], SHAP [17], Integrated Gradients [27], Anchors [23], and ALE [3] designed to explain predictions after a model is trained. However, this after-the-fact approach has proven fundamentally flawed. Research shows that post-hoc explanations can be deliberately manipulated to hide model biases [26], are unstable to minor input changes [12], and frequently produce conflicting or misleading attributions [10, 16, 28], undermining their reliability in high-stakes settings.

This unreliability has shifted focus towards a more robust paradigm: intrinsic interpretability, where models are built to be transparent by design. Architectures like Neural Additive Models (NAMs) [2] and TabNet [4] demonstrate that high performance can be

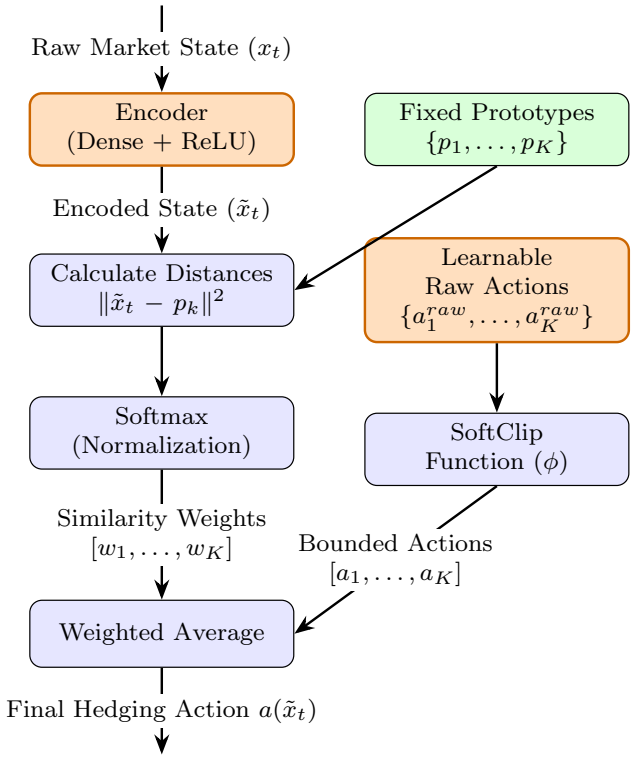


Figure 1: The architecture of the ProtoHedge model. Trainable components are highlighted in orange.

achieved without sacrificing transparency. Among these, concept-based models that reason using prototypes are particularly promising because they mirror human logic. Pioneered by ProtoPNet [11], these models justify decisions with a "this looks like that" approach, comparing parts of an input to learned, prototypical examples. This core idea has been advanced by a range of works focusing on concept definition [5], prototype clarity [20, 21], continual learning [24], and spatial grounding [9]. This evolution from flawed post-hoc patches to human-centric, prototype-based designs provides a clear path toward building trustworthy and transparent hedging models.

3 Proposed Approach: ProtoHedge

To overcome the opacity of traditional deep hedging models, we propose a novel framework that is intrinsically interpretable by design. The model's policy is constructed from a set of canonical market states, or prototypes, ensuring that every hedging decision is transparent and traceable. This section details the model's architecture, its practical use for interpretation, and the end-to-end training procedure.

3.1 Model Architecture and Decision Process

The model architecture of ProtoHedge is shown in Figure 1. We encode the raw market state at time t to a market state vector $\tilde{x}_t \in \mathbb{R}^d$ using a single dense layer with a ReLU activation function. We apply a single shared affine map (followed by a pointwise nonlinearity) to re-scale heterogeneous features and improve distance conditioning; without this re-weighting, Euclidean distances can be dominated by large-scale inputs. We deliberately avoid deeper encoders to preserve transparency and to keep the similarity metric simple and auditable. The same encoder is applied once to each prototype state, and the resulting encoded representations are cached for use in similarity computations. At each time step, given the encoded market state vector $\tilde{x}_t \in \mathbb{R}^d$, the hedging action $a(\tilde{x}_t) \in \mathbb{R}^m$ produced by ProtoHedge is:

$$a(\tilde{x}_t) = \sum_{k=1}^{K} \text{sim}(\tilde{x}_t, p_k) \cdot \phi(a_k^{\text{raw}}) \tag{4}$$

where $\phi(\cdot)$ is the non-linear SoftClip function that smoothly enforces the environment's trading limits. The architecture is composed of the following key components:

- **Prototypes (p_k):** A fixed set of K vectors, $P = \{p_1, \ldots, p_K\}$, where each $p_k \in \mathbb{R}^d$ is an actual, historical market state selected from the data. These serve as the model's static knowledge base of canonical market scenarios.
- **Raw Action Parameters (a_k^{raw}):** A set of learnable, unbounded vectors. These are the underlying trainable parameters of our model, optimized during the training phase described in Section 3.2.
- **Similarity Score (sim):** A normalized score that measures the resemblance between the current state $\tilde{x}_t$ and each prototype p_k. It is computed using a softmax over the negative squared Euclidean distance:

$$\text{sim}(\tilde{x}_t, p_k) = \frac{\exp\left(-\|\tilde{x}_t - p_k\|^2\right)}{\sum_{j=1}^{K} \exp\left(-\|\tilde{x}_t - p_j\|^2\right)} \tag{5}$$

This architecture is intrinsically interpretable because the final decision is a linear combination of well-defined components. Specifically, it is a weighted average of the bounded actions, $a_k = \phi(a_k^{\text{raw}})$, where the influence of each prototype's bounded action is explicitly quantified by its similarity score.

3.2 Training

The model is trained in a two-phase process: first, the static prototype space is constructed, and second, the prototype actions are learned via end-to-end optimization.

- **Phase 1: Prototype Space Construction.** This step is performed once on the set of market states. It involves first partitioning the state space into K clusters to identify regions of high data density, and then extracting a medoid (a representative data point) from each cluster to serve as a prototype. This ensures that the set of prototypes P is fixed before the main policy learning begins.
- **Phase 2: End-to-End Parameter Optimization.** With the prototypes P held constant, the model is trained to learn the optimal set of unbounded action vectors, $\{a_k^{\text{raw}}\}_{k=1}^{K}$. The final, bounded actions, a_k, are obtained by applying the SoftClip function, i.e., $a_k = \phi(a_k^{\text{raw}})$. The training objective is to maximize a utility metric, such as the Conditional Value-at-Risk (CVaR) of the terminal Profit & Loss (P&L).

The model learns by minimizing a loss function, $\mathcal{L}$, defined as the negative of a chosen utility metric. An optimizer, such as Adam,

uses the gradient of this loss to update the trainable parameters $\{a_k^{\text{raw}}\}_{k=1}^{K}$. In the remainder of this subsection, we derive the analytical gradient.

Applying the chain rule across the time steps of a single trajectory, the gradient of the loss $\mathcal{L}$ with respect to a single unbounded prototype action a_k^{raw} is:

$$\frac{\partial \mathcal{L}}{\partial a_k^{\text{raw}}} = \sum_{t=0}^{T-1} \frac{\partial \mathcal{L}}{\partial a(\tilde{x}_t)} \frac{\partial a(\tilde{x}_t)}{\partial a_k^{\text{raw}}} \tag{6}$$

Here, $\partial \mathcal{L}/\partial a(\tilde{x}_t)$ is the upstream gradient from the loss function. The local gradient, $\partial a(\tilde{x}_t)/\partial a_k^{\text{raw}}$, is first written by expanding the definition of the total action $a(\tilde{x}_t)$:

$$\frac{\partial a(\tilde{x}_t)}{\partial a_k^{\text{raw}}} = \frac{\partial}{\partial a_k^{\text{raw}}} \left(\sum_{j=1}^{K} \text{sim}(\tilde{x}_t, p_j) \cdot a_j \right) \tag{7}$$

Because the derivative of a sum is the sum of the derivatives and the similarity scores are constant with respect to the trainable parameters, we can move the derivative operator inside the summation:

$$\frac{\partial a(\tilde{x}_t)}{\partial a_k^{\text{raw}}} = \sum_{j=1}^{K} \text{sim}(\tilde{x}_t, p_j) \frac{\partial a_j}{\partial a_k^{\text{raw}}} \tag{8}$$

Since $a_j = \phi(a_j^{\text{raw}})$ and the raw actions $\{a_j^{\text{raw}}\}$ are independent parameters, the derivative $\partial a_j/\partial a_k^{\text{raw}}$ is non-zero only when $j = k$. This can be expressed using the Kronecker delta, δ_{jk}, as $\partial a_j/\partial a_k^{\text{raw}} = \phi'(a_k^{\text{raw}}) \cdot \delta_{jk}$, where $\phi'(\cdot)$ is the derivative of the SoftClip function. The summation therefore reduces to a single term:

$$\frac{\partial a(\tilde{x}_t)}{\partial a_k^{\text{raw}}} = \text{sim}(\tilde{x}_t, p_k) \cdot \phi'(a_k^{\text{raw}}) \tag{9}$$

Substituting this result back into the chain rule (Equation 6) yields the final form of the gradient:

$$\frac{\partial \mathcal{L}}{\partial a_k^{\text{raw}}} = \left(\sum_{t=0}^{T-1} \frac{\partial \mathcal{L}}{\partial a(\tilde{x}_t)} \cdot \text{sim}(\tilde{x}_t, p_k) \right) \cdot \phi'(a_k^{\text{raw}}) \tag{10}$$

This gradient has a clear interpretation: the update signal for an unbounded action, a_k^{raw}, is the sum of upstream error signals weighted by its prototype's similarity, which is then scaled by the local gradient of the 'SoftClip' function. This final term moderates the update, applying smaller adjustments when the raw action is far into the flat regions of the clip function, ensuring stable learning.

As with the original Deep Hedging framework, the optimization problem is non-convex. Prototype actions are initialized from a random normal distribution, while prototype states are fixed once during clustering. Although local minima cannot be ruled out, in practice we found training to be stable: repeated runs with different random seeds produced very similar hedging profiles and nearly identical utility values. This indicates that, while the optimization landscape is non-convex, it does not materially hinder convergence in our setting.

3.3 Implementation

Our framework is implemented as a custom TensorFlow layer that encapsulates the logic for transparent decision-making. During the forward pass, an input state vector is first standardized using precomputed mean and standard deviation parameters derived from the training data. The model then calculates similarity scores based on a weighted squared Euclidean distance between the standardized input and the fixed prototypes. These distances are converted into a normalized probability distribution using a softmax function, and the final hedging action is efficiently computed via a single matrix multiplication between these similarity weights and the learned prototype actions.

The trainable parameters are the actions corresponding to each prototype, which are initialized as an unbounded tensor from a random normal distribution. To ensure the model's outputs are realistic, these unbounded actions are passed through a `SoftClip` function during the forward pass. This mechanism smoothly enforces the environment's pre-defined trading limits while maintaining stable gradients necessary for effective optimization. This architectural design makes our model a self-contained, differentiable module that can be readily integrated into standard deep hedging training loops.

4 Performance and Interpretability Evaluation

4.1 Experimental Setup

To validate our proposed architecture, we evaluate it in two canonical yet distinct simulated environments, following the setup of Buehler et al. [8]. This two-pronged approach allows us to both verify the model's correctness against a known optimum and test its performance in a more realistic, incomplete market setting.

- **Black–Scholes Environment:** This serves as a critical sanity check. The agent hedges a European call option by trading the underlying spot asset S_t, whose dynamics follow geometric Brownian motion:

$$dS_t = \mu S_t \, dt + \sigma S_t \, dW_t. \tag{11}$$

  In this complete market, the existence of an analytical solution provides a ground truth to confirm our model learns correct hedging principles.
- **Stochastic Volatility Environment:** This presents a more challenging and realistic scenario. The agent trades both the spot asset and an at-the-money (ATM) European call option where drift and volatility are themselves stochastic mean-reverting processes:

$$\begin{aligned} dS_t &= \mu_t S_t \, dt + \sigma_t S_t \, dW_t, \\ d\mu_t &= -\kappa_\mu(\mu_t - \bar{\mu}) \, dt + \xi_\mu \, dW_t^\mu, \\ d\sigma_t &= \kappa_\sigma(\bar{\sigma} - \sigma_t) \, dt + \xi_\sigma \, dW_t^\sigma. \end{aligned} \tag{12}$$

  As an incomplete market with no closed-form solution, this is a standard benchmark for data-driven hedging methods.

Across both environments, we generate independent price paths using the Euler–Maruyama method for discretization. Unless stated otherwise, we align our hyperparameter settings with the original Deep Hedging baseline to allow fair comparison. For example, we set the number of steps per sample to 20 and use CVaR at 50% as the training utility. These choices are not inherent to our approach: alternative hyperparameter values or different state definitions can be used without modifying the model, which highlights the adaptability of our framework to different market settings and risk preferences. The key hyperparameters for training and the environment specifications are summarized in Table 1. For the

market state features, we again follow the Deep Hedging setup: in the Black–Scholes world the state includes the spot price, the current hedge position (delta), and time to maturity, while in the stochastic volatility world it additionally includes the tradable ATM option price and its hedge position (delta).

As is common in prototype-based and case-based reasoning models, the number of prototypes K is a critical hyperparameter that governs the trade-off between model granularity and generalization risk [1, 11]. A small K may fail to capture the diversity of market scenarios, while an overly large K can lead to overfitting and increased computational cost. Following standard machine learning methodology, we determined the optimal values for K through cross-validation on our simulated data. Based on this analysis, we set $K = 100$ for the Black-Scholes setup and $K = 500$ for the more complex stochastic volatility setup.

Table 1: Summary of hyperparameter and configuration

Variable	Black-Scholes	Stochastic Volatility
Training Epochs	800	800
Batch Size	32	32
Training Samples	10,000	10,000
Validation Samples	1,000	1,000
Steps per Sample	20	20
Time Step Size (Δt)	0.02 ($\approx$ weekly)	0.02 ($\approx$ weekly)
No. Prototypes (K)	100	500
Soft Clip Smoothness	1.0	1.0
Spot Trade Limits	± 5 units	± 5 units
Option Trade Limits	-	± 5 units
Option Maturity	4	-
Spot Trading Cost	0.0002	0.0002
Option Cost (vega)	-	0.02
Option Cost (price)	-	0.0005
Utility	CVaR @ 50%	CVaR @ 50%
Optimizer	Adam	Adam
Learning Rate	0.001	0.001
Drift (μ)	0 (constant)	0.1 (mean-reverting)
Realized Volatility	0.2 (constant)	Stochastic[a]
Implied Volatility	-	Stochastic[a]
Payoff (Liability)	$-\max(S_T - 1, 0)$	$-\max(S_T - 1, 0)$

[a]In the stochastic volatility setup, the volatilities follow the discretized mean-reverting log-processes: $\log \sigma_{t+1} = \log \sigma_t + \kappa_i(\log \sigma_0 - \log \sigma_t)\Delta t + \xi_i \varepsilon_t - \frac{1}{2}\xi_i^2 \Delta t$, with similar dynamics for drift r_t. Key parameters are $\kappa_r = 2.0, \kappa_i = 0.1, \xi_r = \xi_i = 0.5, \sigma_0 = 0.2$.

4.2 Hedging Performance Results

Our quantitative analysis focuses on a direct comparison with the original black-box Deep Hedging (DH) model [8]. Black-box DH serves as the most direct and relevant benchmark for our primary contribution: replacing the opaque neural network with an intrinsically interpretable model. While several important extensions to the Deep Hedging framework exist, direct quantitative comparisons are infeasible due to fundamental differences in their experimental settings and objectives. For instance, the work of Murray et al. [19] is formulated in continuous-time, which is mathematically incompatible with our discrete-time settings. The No-Transaction Band Network of Imaki et al. [14] introduces a specialized architecture for managing transaction costs, making it difficult to isolate the performance impact of our interpretability mechanism from their cost-control mechanism. Finally, the work of Mikkilä and Kanniainen [18] is validated on empirical market data, whereas our study relies on controlled environments to understand foundational correctness. Given these incompatibilities, the original black-box DH model remains the most appropriate benchmark for this foundational study. Adapting ProtoHedge to these more advanced settings to enable such comparisons is a promising direction for future research, but is beyond the scope of this paper.

We first validate ProtoHedge (PH) in the classical Black-Scholes world, where an optimal analytical hedge provides a ground-truth benchmark. As shown in Figure 2, the terminal Profit and Loss (P&L) of the ProtoHedge (PH) strategy closely tracks the option's payoff curve, mirroring the behavior of the analytical Black-Scholes (BS) hedge. This demonstrates that our model effectively learns to minimize risk. The utility comparison, measured by the Conditional Value-at-Risk for the worst 1% of outcomes (CVaR at 1%), further confirms this. This metric is used to specifically assess the mitigation of extreme tail risk. While the BS hedge is theoretically optimal, ProtoHedge achieves a nearly identical utility, confirming its ability to learn a near-optimal strategy in a complete market.

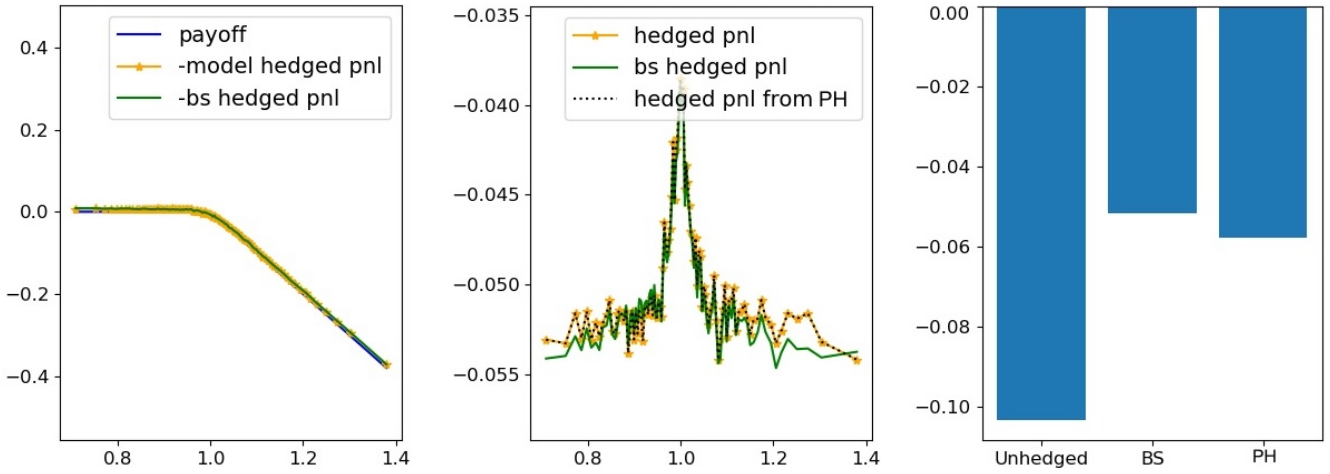
Next, we test ProtoHedge in a more realistic stochastic volatility setting, where no closed-form solution exists. Figure 3 shows that the unhedged portfolio is exposed to significant losses as the option moves in-the-money. In contrast, the ProtoHedge strategy effectively neutralizes this downside risk, maintaining a stable P&L across all terminal spot prices. This result highlights the model's primary function: to reduce risk and minimize variance in complex, unpredictable environments where analytical solutions are unavailable.

Table 2: Absolute and relative expected utility improvements across models in Black–Scholes and Stochastic Volatility settings.

Comparison	Absolute Diff.	Relative Diff.
Black–Scholes		
DH vs Unhedged	+0.04570	+44.0%
ProtoHedge vs Unhedged	+0.04547	+43.8%
ProtoHedge vs BS	−0.00677	−13.1%
ProtoHedge vs DH	−0.00023	−0.40%
Stochastic Volatility		
DH vs Unhedged	+0.03440	+27.6%
ProtoHedge vs Unhedged	+0.03410	+27.4%
ProtoHedge vs DH	−0.00030	−0.33%

Having established our model's correctness and efficacy, we now quantify the performance trade-off associated with its interpretability by comparing it directly to the original black-box Deep Hedging model (DH). Figure 4 compares the expected utility (CVaR at 50%, the training objective) for all strategies in both environments. Table 2 reports the exact utility differences between the models.

A direct comparison reveals that ProtoHedge achieves hedging performance nearly identical to the black-box DH model. In the



(a) Effective terminal payoffs (b) Terminal hedged results (c) Utility comparison (CVaR@1%)

Figure 2: Validation of ProtoHedge (PH) against the analytical Black–Scholes (BS) solution. The plots show effective terminal payoffs, terminal hedged P&L, and a utility comparison (CVaR at 1%).

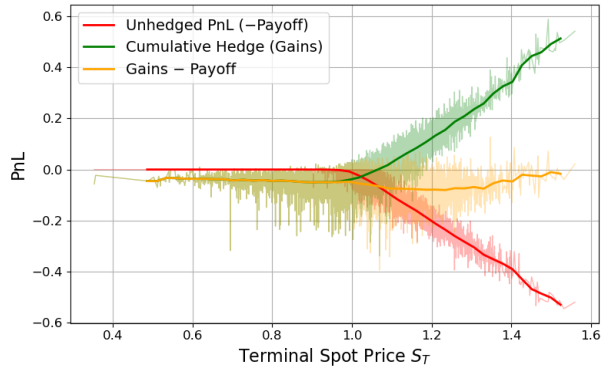
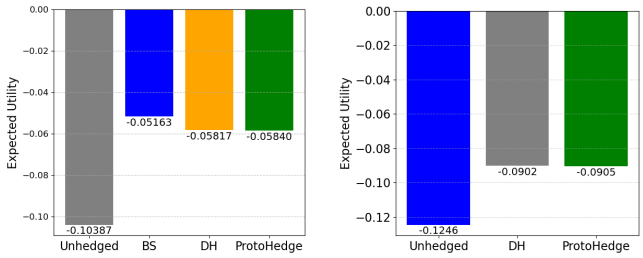


Figure 3: Effective risk reduction: Performance of ProtoHedge vs Unhedged Strategy in the Stochastic Volatility Environment



(a) Black-Scholes Setting (b) Stochastic Volatility Setting

Figure 4: Utility comparison of hedging strategies across both market environments. The plots show the expected utility (CVaR at 50%) for the Unhedged, ProtoHedge, and black-box DH strategies, along with the analytical BS hedge where applicable.

Black-Scholes setting (Figure 4a), the utility difference is a minimal 0.00023, a margin that remains negligible at 0.00030 in the more complex stochastic volatility setting (Figure 4b). This result is crucial: it quantifies the cost of interpretability as exceptionally low. When contextualized by the substantial utility gains both models provide over an unhedged strategy, this minor trade-off underscores our central claim that intrinsic interpretability and high-performance hedging are not mutually exclusive.

Taken together, these analyses provide a clear validation of our interpretable model. ProtoHedge demonstrates its foundational correctness by replicating the optimal analytical solution in a classical Black-Scholes market. Moreover, it proves its practical value in a complex stochastic volatility environment by achieving hedging effectiveness comparable to its black-box counterpart. This confirms that the substantial benefits of explainability–enabling model auditing and building trust–are attainable with only a minimal and quantifiable impact on performance.

4.3 Interpreting a Hedging Decision

To illustrate how our model makes decisions, we analyze a sample trajectory from the Black–Scholes environment. This example shows not only the actions taken by the agent, but also helps explain *why* they are reasonable in light of the evolving market conditions.

In this path, the option ends up in-the-money as the spot price increases steadily above 1.1. Since the agent is short the option, this creates a growing liability that must be hedged. Our model responds by gradually increasing its holdings in the underlying asset, as shown in Figure 5. The delta rises quickly and remains high to offset the upward price movement and limit potential losses.

This behavior is expected and shows that the agent builds up protection early and maintains it as risk grows, demonstrating a clear and intuitive hedging strategy.

To better understand how our model selects actions, we examine the current market state at a specific timestep and compare it with

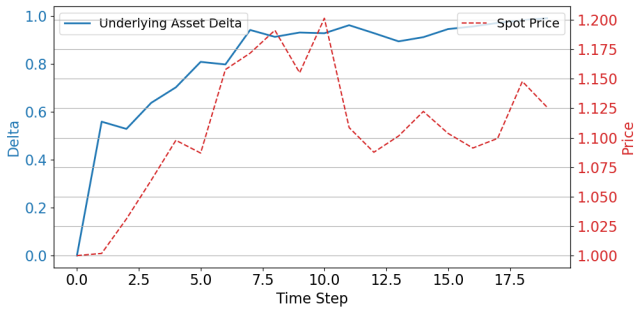


Figure 5: Example path showing the evolution of the spot price and the ProtoHedge's position for the underlying asset.

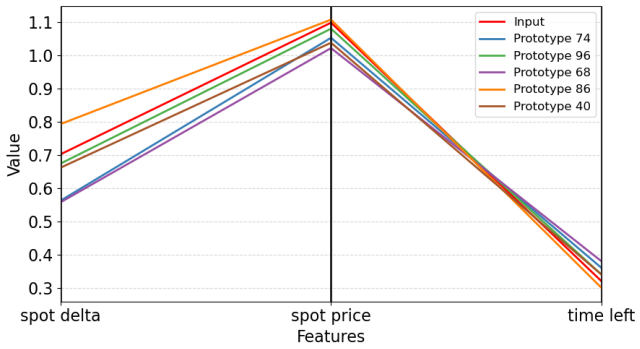


Figure 6: Parallel coordinates plot comparing the current market state (in red) with its 5 closest prototypes in the Black-Scholes setting.

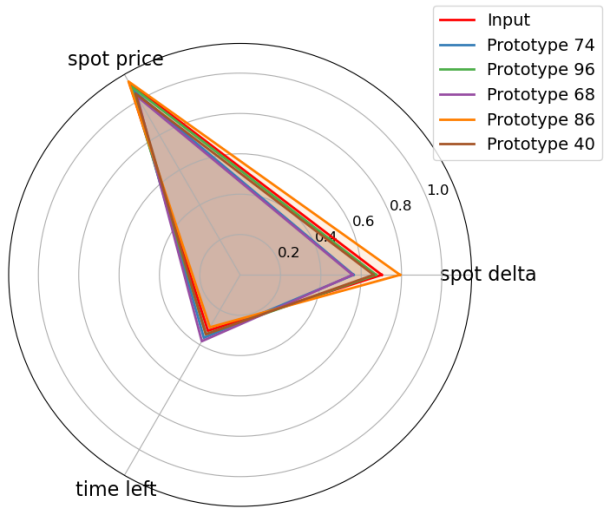


Figure 7: Radar plot of the current market state and its 5 closest prototypes, highlighting similarity in the feature space.

the top five closest prototypes identified by the model. Figures 6 and 7 show this comparison using parallel coordinates and radar plots. The red line represents the current input, while the remaining lines correspond to the nearest prototypes. We observe that the input is visually close to several stored prototypes across all features, which explains why the model confidently applies a similar action. This highlights the interpretability of our approach: since the input resembles known patterns, the model can reuse previously learned behaviors in a transparent and intuitive way.

4.4 Characterizing the Learned Prototype Space

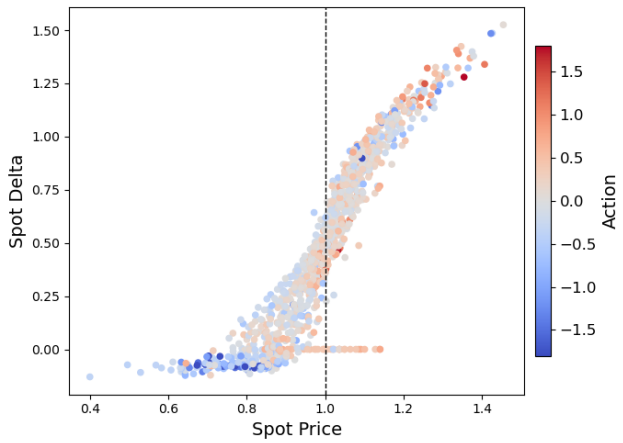
A primary benefit of ProtoHedge is the ability to audit the hedging strategy learned by the model so that the user can build trust. Beyond analyzing individual decisions, we can inspect the entire set of learned prototypes to answer two critical questions: (1) Does the model's knowledge base reflect sound financial principles? and (2) How does it dynamically apply this knowledge?

To investigate the structure of the learned knowledge, we visualize the 500 prototypes from the stochastic volatility environment in Figure 8. The band-like patterns visible in both subplots arise naturally because prototypes cluster along regions of consistent hedging behavior (e.g., near-zero hedge out-of-the-money and positive hedge in-the-money), rather than from redundant prototypes. These plots confirm that ProtoHedge has independently rediscovered fundamental principles of derivatives hedging. Subfigure 8a illustrates the classic "S-curve" relationship between an option's delta and the underlying spot price, with prototypes correctly recommending "sell" actions (blue) for out-of-the-money states and "buy" actions (red) for in-the-money states. Subfigure 8b highlights time-sensitive strategies: as time left decreases, the recommended actions become more decisive (darker colors), consistent with the effects of Theta and Gamma.

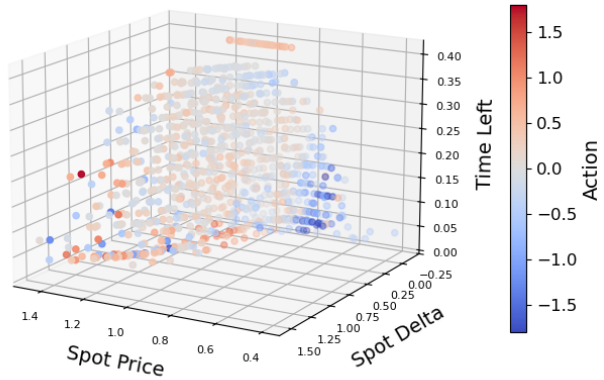
In summary, this global audit provides strong evidence of the model's value. ProtoHedge does not simply produce a hedging decision; it builds a transparent and coherent strategy grounded in sound financial logic. Its learned behavior aligns closely with established principles of derivatives hedging and organizes strategies around a small set of critical market regimes, making it a trustworthy and auditable system.

4.5 Limitations

While our results demonstrate the viability of the prototype-based approach, it is important to acknowledge the deliberate scope of our evaluation and its inherent limitations. First, our experiments were conducted in simulated market environments (Black-Scholes and stochastic volatility) using European-style options. These settings were chosen intentionally as they provide a controlled environment with clear benchmarks, which is essential for validating a novel architecture and proving its ability to learn fundamental hedging principles. However, these simulations do not capture the full complexity of real-world markets, such as price jumps or fat-tailed distributions. Furthermore, the model has not yet been tested on more complex, path-dependent derivatives (e.g., Asian or barrier options), where the state representation would need to be richer. Second, our analysis highlights a trade-off between interpretability



(a) S-curve between price and hedge action learned from data



(b) Actions become more decisive (darker colors) close to expiry

Figure 8: Visualization of the learned prototype space. The plots confirm that ProtoHedge's internal logic, which is learned from data, aligns with fundamental financial principles.

and computational scalability. As noted in Section 3.3, the computational cost during training scales with the number of prototypes, K. Moving from the Black-Scholes (K=100) to the stochastic volatility (K=500) environment required a significant increase in both prototypes and training time. Applying the model to high-dimensional problems, such as hedging a large portfolio of diverse instruments, would pose a substantial computational challenge that is beyond the scope of this initial study. These limitations define clear and important directions for future research, which we outline in the following section.

5 Conclusion and Future Work

The practical adoption of deep hedging is limited by the black-box nature of its underlying neural networks, which poses challenges for trust, auditability, and regulatory oversight. In this work, we introduce an intrinsically interpretable hedging framework to address this limitation. Our model is transparent by design, making hedging decisions by computing the similarity between the current market state and a learned set of representative prototypes. The final action is a weighted combination of simple policies associated with these prototypes, making each decision fully traceable.

Our experimental results demonstrate the effectiveness of this approach. First, the model successfully replicates the optimal analytical solution in a complete Black-Scholes market, confirming its ability to learn correct hedging strategies. Second, its performance is comparable to that of a conventional black-box deep hedging model in a more realistic stochastic volatility environment. Our results suggest that the integration of intrinsic interpretability into deep hedging systems is achievable without a significant compromise in performance, which is a crucial step towards their practical deployment in the financial industry.

Future work will proceed along three main axes. First, empirical validation: a crucial next step is to test the framework on historical market data for a variety of instruments, including path-dependent options (e.g., Asian or barrier options). This will test the model's robustness against real-world dynamics like market frictions and non-stationarity. Since our current design is a proof-of-concept, training on empirical data will also allow us to backtest whether the learned prototypes correspond to recognizable market scenarios, such as financial crises or flash crashes, and whether the model responds to them in a consistent and interpretable way. In parallel, engaging professional traders to backtest and assess the model in practice would provide valuable external validation and highlight its usability in real trading contexts. Second, scalability and optimization: we will systematically investigate the trade-off between the number of prototypes, feature space dimensionality, and computational performance. This includes exploring more efficient methods for prototype selection and distance calculation, such as approximate nearest-neighbor search, which will be essential for applying this interpretable framework to more complex, high-dimensional hedging problems. Third, comparative evaluation: it will be important to benchmark our approach against other recent deep hedging models, such as Deeper Hedging under Chiarella–Heston dynamics [13], to better assess its strengths and limitations under more realistic market conditions.

Acknowledgments

We thank the anonymous reviewers for their thorough evaluation and valuable suggestions. We are also grateful to Dr Pedro Mediano (Imperial College London) for feedback following the submission. This work is supported by the United Kingdom EPSRC (grant numbers UKRI256, EP/V028251/1, EP/N031768/1, EP/S030069/1, and EP/X036006/1), Altera, Intel, AMD, SiliconFlow, and Google Cloud.

References

[1] Agnar Aamodt and Enric Plaza. 1994. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications* 7, 1 (1994), 39–59.

[2] Rishabh Agarwal, Levi Melnick, Nicholas Frosst, Xuezhou Zhang, Ben Lengerich, Rich Caruana, and Geoffrey E. Hinton. 2020. Neural Additive Models: Interpretable Machine Learning with Neural Nets. In *Advances in Neural Information Processing Systems*, Vol. 33. Curran Associates, Inc., Virtual Event, 4699–4711.

[3] Daniel W. Apley and Jingyu Zhu. 2020. Visualizing the Effects of Predictor Variables in Black Box Supervised Learning Models. *Journal of the Royal Statistical*

Society: Series B (Statistical Methodology) 82, 4 (2020), 1059–1086. doi:10.1111/rssb.12377

[4] Sercan Ö. Arık and Tomas Pfister. 2021. TabNet: Attentive Interpretable Tabular Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. AAAI Press, Virtual Event, 6679–6687.

[5] Mohammad Asadi, Vinitra Swamy, Jibril Frej, Julien Vignoud, Mirko Marras, and Tanja Käser. 2023. RIPPLE: Concept-Based Interpretation for Raw Time Series Models in Education. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37. AAAI Press, Washington, DC, USA, 15903–15911. doi:10.1609/aaai.v37i13.26888

[6] Alceu Bissoto, Michel Fornaciali, Eduardo Valle, and Sandra Avila. 2019. (De)Constructing Bias on Skin Lesion Datasets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, Long Beach, CA, USA, 2766–2774. doi:10.1109/CVPRW.2019.00335

[7] Bartosz Brożek, Michał Furman, Marek Jakubiec, and Bartłomiej Kucharzyk. 2024. The black box problem revisited. Real and imaginary challenges for automated legal decision making. *Artificial Intelligence and Law* 32, 2 (2024), 427–440. doi:10.1007/s10506-023-09356-9

[8] H. Buehler, L. Gonon, J. Teichmann, and B. Wood. 2019. Deep Hedging. *Quantitative Finance* 19, 8 (2019), 1271–1291. doi:10.1080/14697688.2019.1571683

[9] Zachariah Carmichael, Suhas Lohit, Anoop Cherian, Michael J. Jones, and Walter J. Scheirer. 2024. Pixel-Grounded Prototypical Part Networks. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. IEEE, Waikoloa, HI, USA, 4756–4767. doi:10.1109/WACV57701.2024.00470

[10] Zachariah Carmichael and Walter J. Scheirer. 2021. A Framework for Evaluating Post Hoc Feature-Additive Explainers.

[11] Chaofan Chen, Oscar Li, Chaofan Tao, Alina Jade Barnett, Jonathan Su, and Cynthia Rudin. 2018. This Looks Like That: Deep Learning for Interpretable Image Recognition. In *Advances in Neural Information Processing Systems*, Vol. 31. Curran Associates, Inc., Montreal, Canada, 8990–9001.

[12] Ann Kathrin Dombrowski, Maximilian Alber, Christopher J. Anders, Marcel Ackermann, Klaus-Robert Müller, and Pan Kessel. 2019. Explanations can be manipulated and geometry is to blame. In *Advances in Neural Information Processing Systems*, Vol. 32. Curran Associates, Inc., Vancouver, Canada, 9726–9737.

[13] Kang Gao, Stephen Weston, Perukrishnen Vytelingum, Namid Stillman, Wayne Luk, and Ce Guo. 2023. Deeper Hedging: A New Agent-based Model for Effective Deep Hedging. In *4th ACM International Conference on AI in Finance (ICAIF '23)*. ACM, New York, NY, USA, 270–278. doi:10.1145/3604237.3626913

[14] Shota Imaki, Kentaro Imajo, Katsuya Ito, Kentaro Minami, and Kei Nakagawa. 2023. No-Transaction Band Network: A Neural Network Architecture for Efficient Deep Hedging. *The Journal of Financial Data Science* 5, 2 (2023), 84–99.

[15] Petter Kolm and Gordon Ritter. 2019. Dynamic Replication and Hedging: A Reinforcement Learning Approach. *The Journal of Financial Data Science* 1 (01 2019), 159–171. doi:10.3905/jfds.2019.1.1.159

[16] Satyapriya Krishna, Tessa Han, Alex Gu, Javin Pombra, Shahin Jabbari, Steven Wu, and Himabindu Lakkaraju. 2022. The Disagreement Problem in Explainable Machine Learning: A Practitioner's Perspective. arXiv preprint arXiv:2202.01602.

[17] Scott M. Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems*, Vol. 30. Curran Associates, Inc., Long Beach, CA, USA, 4765–4774.

[18] Oskari Mikkilä and Juho Kanniainen. 2023. Empirical deep hedging. *Quantitative Finance* 23, 1 (jan 2023), 111–122. doi:10.1080/14697688.2022.2136037

[19] Phillip Murray, Ben Wood, Hans Buehler, Magnus Wiese, and Mikko Pakkanen. 2022. Deep hedging: Continuous reinforcement learning for hedging of general portfolios across multiple risk aversions. In *Proceedings of the Third ACM International Conference on AI in Finance*. Association for Computing Machinery, New York, NY, USA, 361–368.

[20] Meike Nauta, Jörg Schlötterer, Maurice van Keulen, and Christin Seifert. 2023. PIP-Net: Patch-Based Intuitive Prototypes for Interpretable Image Classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, Vancouver, Canada, 2744–2753. doi:10.1109/CVPR52729.2023.00269

[21] Mateusz Pach, Dawid Rymarczyk, Koryna Lewandowska, Jacek Tabor, and Bartosz Zieliński. 2024. LucidPPN: Unambiguous Prototypical Parts Network for User-centric Interpretable Computer Vision. arXiv preprint arXiv:2405.14331.

[22] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*. Association for Computational Linguistics, San Diego, California, 97–101. doi:10.18653/v1/N16-3020

[23] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Anchors: High-Precision Model-Agnostic Explanations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32. AAAI Press, New Orleans, Louisiana, 1527–1535. doi:10.1609/AAAI.V32I1.11491

[24] Dawid Rymarczyk, Joost Van De Weijer, Bartosz Zieliński, and Bartłomiej Twardowski. 2023. ICICLE: Interpretable Class-Incremental LEarning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. IEEE, Paris, France, 1887–1898. doi:10.1109/ICCV51070.2023.00181

[25] Khaled Saab, Sarah Hooper, Mayee Chen, Michael Zhang, Daniel Rubin, and Christopher Ré. 2022. Reducing Reliance on Spurious Features in Medical Image Classification with Spatial Specificity. In *Proceedings of Machine Learning Research (Proceedings of Machine Learning Research, Vol. 162)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (Eds.). PMLR, Baltimore, Maryland, USA, 18728–18753.

[26] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. 2020. Fooling LIME and SHAP: Adversarial Attacks on Post hoc Explanation Methods. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. Association for Computing Machinery, Barcelona, Spain, 180–186. doi:10.1145/3375627.3375830

[27] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic Attribution for Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, Sydney, Australia, 3319–3328.

[28] Vinitra Swamy, Jibril Frej, and Tanja Käser. 2023. The future of human-centric eXplainable Artificial Intelligence (XAI) is not post-hoc explanations. arXiv preprint arXiv:2307.00364.

[29] Edoardo Vittori, Michele Trapletti, and Marcello Restelli. 2020. Option Hedging with Risk Averse Reinforcement Learning. arXiv:2010.12245 [q-fin.TR] https://arxiv.org/abs/2010.12245

[30] Hanhui Xu and Kyle Michael James Shuttleworth. 2024. Medical artificial intelligence and the black box problem: a view based on the ethical principle of "do no harm". *Intelligent Medicine* 4, 1 (feb 2024), 52–57. doi:10.1016/j.imed.2023.08.001