Entropy-Guided Counterexample Refinement

Tilted Risk, Predicate Trees, and Certified Path Blocking

Anonymous 11 November 2026

Abstract. A spurious counterexample supplies two disjoint sets of concrete states at each cut, the endpoints of feasible prefixes and the starts of feasible suffixes. We derive a pivot score by assigning these states opposite labels and refitting a label distribution in each abstraction cell. For positive exponential tilt of log loss, the optimal value is Arimoto conditional Rényi entropy. At unit temperature the ranking reduces to a sum of geometric means of prefix and suffix masses, the square-root impurity used in decision-tree boosting. A separating predicate extends to a least layered certificate at a specified cut, which excludes every refined descendant of the rejected path. The certificate argument extends to internal truth-value observations in an infinity-topos. Lean verifies the integer scoring bounds, the zero-score separation criterion, and the completion checker. A reproducible finite study isolates the effects of pivot order from those of the refinement policy.

Keywords. CEGAR, decision-tree boosting, Arimoto entropy, tilted risk, predicate abstraction, Lean, higher topos.

1 Introduction

Counterexample-guided abstraction refinement (CEGAR) checks a simplified transition system and analyzes any candidate counterexample against the concrete system [3, 5]. When that candidate is spurious, its failure gives evidence about which concrete states should be distinguished. The verifier must choose a distinction that can be represented compactly and prove that it addresses the path. Predicate abstraction and interpolation provide established ways to obtain such distinctions [8, 9, 13].

We treat the choice of a distinction as a finite weighted classification problem. At a cut of the chosen abstract path, concrete states supporting a prefix form one class, and states supporting a suffix form the other. They are disjoint exactly when the selected path has no concrete lift. A candidate observation predicate partitions these examples. It can be ranked by its ability to separate the two classes, and several predicates can be composed as a decision tree.

The classification problem uses the same exponentially tilted empirical risk studied in machine learning [11]. Optimizing that objective over distributions constant on each observation cell gives an exact information measure. At unit temperature its reduction has the same ordering as the geometric-mean impurity analyzed for decision-tree boosting by Kearns and Mansour [10]. The classification classes here come from an exact path query, and a verifier can check the logical consequences of the chosen predicates.

Contributions. First, we derive the conditional entropy of the optimal cell predictor and identify its unit-temperature criterion as a geometric mean of incompatible witness masses. We give a local weak-learning bound on its reduction, and an example in which every individual predicate has zero gain but a decision tree separates the classes.

Second, we prove a completion theorem for a separator at one cut. It constructs the least sequence of layered predicates with the prescribed cut predicate. Refinement by the entire sequence removes every descendant of the chosen path. The higher-categorical form uses observations into the subobject classifier and obtains elimination by a map from the residual descendant object to the initial object.

Third, we supply two executable components. Lean verifies natural-number brackets for a pivot score, a certified comparison bound, separation at zero score, and the completion of a valid finite prefix certificate. A separate reference verifier implements exact integer ranking and the complete finite CEGAR loop. A controlled comparison records model checks, quotient sizes, and learned tree sizes under four policies.

Relation to boosting and prior verification work. The common optimization is a greedy choice from a dictionary of features. AdaBoost and related coordinate methods choose features through a weighted loss or gradient on examples [6, 4]. In the model considered here, prediction probabilities are freely refitted on each cell of a refined partition. This hypothesis is used in the exact entropy identity. Decision-tree learning with implication counterexamples already uses entropy to guide inductive invariant inference [7]. The present construction uses the exact prefix and suffix of a fixed trace, a tilted loss, and a certificate that eliminates all of its refined descendants. It develops the observation and path-fiber semantics of [1].

The manuscript and companion were developed with AI assistance. The code and the recorded checks are identified by file and theorem name. The general results are proved in the text.

2 Exact evidence at a cut

Let X be a finite set of concrete states. Let $I, B \subseteq X$ be the initial and bad states, and $T_\ell \subseteq X \times X$ the relation for label ℓ. An observation $q : X \to A$ induces a may

graph on its realized keys. Its boundary predicates and edge relations are the images of the concrete ones. Hence every concrete bad path projects to an abstract bad path. A model check on the abstract graph can therefore prove safety when no abstract bad path exists.

A candidate bad path has keys $a_0, \dots, a_m$ and edge labels $\ell_1, \dots, \ell_m$. Write $A_i = q^{-1}(a_i)$, and $\mathrm{post}_\ell(S) = \{y : \exists x \in S,\ xT_\ell y\}$. Define forward and backward sets by

$$F_0 = I \cap A_0, \qquad F_i = A_i \cap \mathrm{post}_{\ell_i}(F_{i-1}), \tag{1}$$

$$D_m = B \cap A_m, \quad D_i = \{x \in A_i : \exists y \in D_{i+1},\ xT_{\ell_{i+1}} y\}. \tag{2}$$

Each set can be computed by a Boolean dynamic program for the given path. The same definitions apply to symbolic predicates whenever the images or a sound proof of the relevant implication can be constructed.

Lemma 2.1 (Cut criterion)**.** *At each cut i, the set $F_i \cap D_i$ is the set of states occurring at position i in some concrete lift of the selected bad path.*

Proof. An induction forward shows that each state in F_i has a concrete prefix with the specified observations and labels. An induction backward gives the suffix statement for D_i. At a common state the two paths concatenate, using the common occurrence only once. Each lift restricts to a prefix and a suffix. □

The two sets are therefore disjoint for every cut of a spurious path. In a useful cut both sets are inhabited, so their union is a finite classification problem with opposite labels. On a cut with one empty side, a constant predictor already has zero classification loss. An exact certificate can use such a cut even when it provides no training examples of one class.

2.1 Observed collisions

For a proposed observation refinement q', with $q = rq'$, each old cell is a union of new cells. At cut i, each new cell C can contain both a state in F_i and a state in D_i. This is an observed collision of prefix and suffix evidence. A predicate p can be used as an extra coordinate $q' = (q, p)$, with factor map pr_1. A decision tree in a finite dictionary of Boolean predicates defines a single composite Boolean predicate by labeling its leaves.

Example 2.2 (Independent middle witnesses)**.** Take states s, u, v, t, initial state s, bad state t, and edges $s \to u$ and $v \to t$. Let u, v share a cell M. The may graph has a path from the key of s through M to the key of t. At its middle position, $F_1 = \{u\}$ and $D_1 = \{v\}$. A predicate that distinguishes u from v eliminates this collision. With larger graphs, the same separation at one position must be propagated through the other positions to exclude every descendant. Section 5 gives a constructive propagation law.

For scoring, choose positive weights on all states in $F_i \amalg D_i$, normalized to a probability law μ. Let $Y = 1$ on F_i and $Y = 0$ on D_i. For several sampled obligations, tag each example by its path and cut before forming the disjoint union. This permits the same state to support different roles in different obligations. The numerical analysis below uses a fixed measure and fixed labels for each comparison.

2.2 Abstract paths and actual decisions

A candidate with a concrete lift supplies a true violation, including its concrete state sequence. A rejected candidate justifies a finer observation with a strict increase in the number of realized keys. Once this increase is shown, the finite carrier bounds the number of rounds. This is a distinct argument from any rate of decrease of the entropy on a single cut. The complete decision procedure below combines the two arguments by producing a certified refinement for every rejected candidate.

3 Tilted loss and conditional entropy

Fix a finitely supported joint law of (X, Y) with finite label set $\mathcal{Y}$. Write $Q = q(X)$ for a finite observation. For each realized cell a, let $\pi_{a,y} = \Pr_\mu(Y = y \mid Q = a)$. A cell predictor chooses a probability vector $s_a \in \Delta(\mathcal{Y})$. The per-example log loss is $\ell_s(X, Y) = -\log s_Q(Y)$. We use natural logarithms throughout. Let $\tau > 0$. Define the fitted tilted loss

$$\mathcal{L}_\tau(q) = \inf_s \tau \log \mathbb{E}_\mu \exp\big(\ell_s(X, Y)/\tau\big). \tag{3}$$

Only labels with positive conditional mass enter each inner expectation. Predictors assigning zero to such a label have infinite loss. A boundary minimizer can instead be read as an infimum over strictly positive probability vectors.

The variational identity

$$\tau \log \mathbb{E}_\mu e^{\ell/\tau} = \sup_{\nu \ll \mu} \{\mathbb{E}_\nu \ell - \tau D_{\mathrm{KL}}(\nu \| \mu)\} \tag{4}$$

follows by taking the KL divergence to the probability measure with density proportional to $e^{\ell/\tau}$. For a uniform μ, the objective on the right is expected loss plus τ times the Shannon entropy of the weights, up to an additive constant. Thus the tilt defines a maximum-entropy weighting of examples, and fitting a cell predictor is a subsequent minimization. See [11] for the general tilted-risk setting.

Theorem 3.1 (Exact cell refit)**.** *Let $\alpha = \tau/(1+\tau)$. Then*

$$\begin{aligned} \mathcal{L}_\tau(q) &= H^{\mathrm{A}}_\alpha(Y \mid Q) \\ &= \tau \log \sum_a \mu(Q = a) \left(\sum_{y \in \mathcal{Y}} \pi^\alpha_{a,y} \right)^{1/\alpha}, \end{aligned} \tag{5}$$

where H^{A}_{α} is Arimoto conditional Rényi entropy [16, 17].

Proof. The expectation inside the logarithm in (3) separates by cell. On the positive support of π_a, put $k = 1/\tau$. The problem for one cell is minimizing $\sum_y \pi_{a,y} s_a(y)^{-k}$ over a simplex. Allocating mass outside the support can only increase the terms on the support, so an optimum assigns it zero mass. On the positive simplex each summand is strictly convex, and it diverges on approaching zero in its own coordinate. The Lagrange multiplier condition gives $\pi_{a,y} s_a(y)^{-k-1}$ constant on the support. Hence

$$s^\star_a(y) = \frac{\pi^\alpha_{a,y}}{\sum_z \pi^\alpha_{a,z}}, \qquad \alpha = \frac{1}{k+1}.$$

For one supported label the same formula assigns it probability one. Substitution gives the minimum $(\sum_y \pi^\alpha_{a,y})^{1/\alpha}$. Finally $\alpha/(1-\alpha) = \tau$, which is the coefficient in the Arimoto definition. □

Corollary 3.2 (Refinement gain)**.** *For $q' = (q, p)$, choosing p to maximize*

$$\Delta_\tau(p \mid q) = H^{\mathrm{A}}_{\alpha}(Y \mid q(X)) - H^{\mathrm{A}}_{\alpha}(Y \mid q(X), p(X))$$

is equivalent to choosing the best attainable reduction of the tilted log loss after a separate refit on every child cell.

Proof. The parent predictor class embeds in the refined predictor class by making each child inherit its parent's probabilities. Apply Theorem 3.1 to the two optima. □

The gain is nonnegative. It vanishes precisely when every positive-mass child has the same conditional label distribution as its parent. Indeed, equality means an optimal parent predictor is also optimal in every child of positive mass. The unique conditional minimizer above determines π_a from its coordinates by raising them to $1/\alpha$ and renormalizing. The converse follows by using the same conditional minimizers.

3.1 Two limiting temperatures

As $\tau \to \infty$, $\alpha \uparrow 1$. For each cell,

$$\log \left(\sum_y \pi^\alpha_{a,y} \right)^{1/\alpha} = (1-\alpha) H(\pi_a) + o(1-\alpha).$$

This follows by differentiating the sum at $\alpha = 1$ on the positive support. Since there are finitely many cells and $\tau(1-\alpha) = \alpha$, (5) tends to conditional Shannon entropy. The gain tends to $I(Y; p(X) \mid Q)$. This is the weighted log-loss form of the decision-tree information gain of [14].

As $\tau \downarrow 0$, $\alpha \downarrow 0$. Put $k_a = |\{y : \pi_{a,y} > 0\}|$. Each sum of powers tends to k_a. The log of the weighted sum of positive terms is bounded above by the log of the largest term and below by it plus the log of its positive cell weight. Multiplying these bounds by τ, and using $\tau/\alpha = 1 + \tau$, gives

$$\lim_{\tau \downarrow 0} \mathcal{L}_\tau(q) = \max_{a : \mu(Q=a) > 0} \log k_a.$$

In the binary case this limit detects whether any mixed cell remains. At finite temperature the score also distinguishes different class proportions and cell masses.

3.2 Unit temperature

For binary labels and $\tau = 1$, write $f_C = \mu(F \cap C)$ and $d_C = \mu(D \cap C)$ for an observation cell. Since $(\sqrt{u} + \sqrt{1-u})^2 = 1 + 2\sqrt{u(1-u)}$,

$$\mathcal{L}_1(q) = \log(1 + 2\Psi(q)), \qquad (6)$$
$$\Psi(q) = \sum_C \sqrt{f_C d_C}.$$

Thus the score for a binary split of one cell is just the sum of two square roots. For a fixed witness law, the score is zero exactly when every observation cell is pure. With strictly positive weight on each prefix and suffix witness, this is equivalent to their having disjoint images under the observation.

Equal size of the children does not ensure any reduction. Four equally weighted examples with two of each label can be split into two balanced mixed children, preserving Ψ, or into two pure children, making $\Psi = 0$. The labels are essential to the pivot score.

4 Local gain and the boosting condition

Consider a mixed observation cell C, with positive masses f_C, d_C. For a Boolean predicate p write

$$u = \Pr(p = 1 \mid F, C), \qquad v = \Pr(p = 1 \mid D, C).$$

The two probabilities describe the effect of the pivot on the two witness classes. If the classes are balanced by reweighting each to probability $1/2$, the best classifier that predicts from p, up to reversing its labels, has error $1/2 - |u - v|/2$. This is the local weak advantage used in the following estimate.

Proposition 4.1 (Overlap reduction)**.** *The contribution $\sqrt{f_C d_C}$ after splitting C is multiplied by*

$$\rho(u, v) = \sqrt{uv} + \sqrt{(1-u)(1-v)}.$$

The relative decrease is at least $\frac{1}{2}(u - v)^2$.

Proof. The two child masses are respectively $(f_C u, d_C v)$ and $(f_C(1-u), d_C(1-v))$. Taking the geometric mean in each child and adding gives the factor ρ. The dot

product of $(\sqrt{u}, \sqrt{1-u})$ with $(\sqrt{v}, \sqrt{1-v})$ is ρ. Their determinant has square $1-\rho^2$. Also

$$\begin{aligned}|u-v| &= |\sqrt{u(1-v)} - \sqrt{v(1-u)}| \\ &\quad \cdot (\sqrt{u(1-v)} + \sqrt{v(1-u)}) \\ &\le \sqrt{1-\rho^2},\end{aligned}$$

because the second factor is at most one by Cauchy–Schwarz. Therefore $1-\rho \ge (1-\rho^2)/2 \ge (u-v)^2/2$. □

This is the usual overlap of two Bernoulli laws, also called their Bhattacharyya coefficient. Its use here depends on the labels induced by the trace. Equation (6) is the same ordering as the weighted square-root impurity of binary decision trees [10, 2]. If a weak predicate achieves balanced classification error at most $1/2-\gamma$, it reduces the contribution of the selected cell by at least a factor $2\gamma^2$ of that contribution. The amount of global improvement depends on the share of the total score in that cell.

Corollary 4.2 (Contraction under a uniform advantage)**.** *Fix the labeled witness set and its weights. Suppose each selected mixed cell carries at least a fraction $\beta > 0$ of the current potential, and the proposed pivot has $|u-v| \ge \eta > 0$ in that cell. After k such splits,*

$$\Psi_k \le (1-\beta\eta^2/2)^k \Psi_0.$$

If the least positive witness mass is w_*, purity follows once the displayed upper bound is strictly less than w_*.

Proof. Only the selected cell changes. Apply Proposition 4.1 and multiply the per-round inequalities. Every mixed cell has at least one witness from each class. Its contribution is at least $\sqrt{w_* w_*} = w_*$. □

The hypotheses describe a single separation problem. During CEGAR the chosen path and even the labels on witnesses can change. Summing potentials of successively chosen paths does not give this contraction without a common witness law and a rule preserving the potential.

4.1 Zero gain with a useful feature

Consider $X = \{0,1\}^2$ with uniform mass and $Y = X_1 \oplus X_2$. Each coordinate has the same conditional label law in its two children as at the root. Both coordinates therefore have zero gain at every positive temperature at the root. After either first split, the second coordinate separates the classes in each child.

For the set function $G(S)$ given by the improvement after observing the coordinates in S, this yields $G(\{1\}) = G(\{2\}) = 0$ and $G(\{1,2\}) = \log 2$. It violates the diminishing-returns inequality of submodularity. A policy requiring strict gain at each split fails on this dictionary. A policy requiring strict division of the witnesses can advance through the zero-gain step. Higher-order features can instead be included as individual predicates, at a cost in dictionary size.

4.2 Greedy separator trees

Let $\mathcal{P}$ be a finite dictionary of Boolean predicates on A_i. At a mixed node choose a predicate that sends witnesses to both children. Among these, minimize the two-child potential, with a fixed tie rule. Recurse on each child, labeling pure leaves by their unique observed class. Every recursive node inherits its entire state region, including points with no witness label. This gives a total Boolean predicate on A_i.

Proposition 4.3 (Exact separation by signatures)**.** *A tree over $\mathcal{P}$ separates F_i from D_i exactly when the two sets have disjoint images under their full predicate signatures. In that case the greedy recursion above succeeds, uses at most $|F_i| + |D_i|$ leaves, and has at most $|\mathcal{P}|$ tests on any root-to-leaf path.*

Proof. Two witnesses with identical signatures follow identical branches in every tree. Conversely, at a mixed node with different class signatures, some coordinate distinguishes at least one opposite-class pair. Therefore at least one admissible split exists. Every split divides the witness set into two proper nonempty subsets, so recursion terminates with at most one leaf per witness. A tested predicate is constant in each of its children and cannot be used again along that path. □

With $n = |A_i|$ and $d = |\mathcal{P}|$, a simple implementation scans all d predicates and their witness values at each of at most $n-1$ internal nodes. A coarse bound is $O(dn^2)$ Boolean and counting operations, ignoring the bit cost of integer multiplication. The tree can be compact even if its fully enumerated truth table is large. The procedure can return failure on a dictionary with a conflicting signature, in which case exact forward predicates still supply a certificate.

5 Completing a separator into a certificate

Fix the path and its sets from Section 2. A layered certificate is a sequence $J_i \subseteq A_i$ with

$$I \cap A_0 \subseteq J_0, \tag{7}$$

$$A_j \cap \mathrm{post}_{\ell_j}(J_{j-1}) \subseteq J_j \qquad (1 \le j \le m), \tag{8}$$

$$J_m \cap B = \varnothing. \tag{9}$$

Such a sequence is an inductive proof along the selected word. Its intermediate predicates need not hold at the corresponding state location on every execution of the system. They are used only for the candidate with these observations and labels.

Theorem 5.1 (Pinned completion)**.** *Fix i and a predicate $K \subseteq A_i$. There exists a layered certificate with $J_i = K$ exactly when*

$$F_i \subseteq K \quad \text{and} \quad K \cap D_i = \varnothing.$$

When these conditions hold, the following is its coordinatewise least completion:

$$J_j = \begin{cases} F_j, & j < i, \\ K, & j = i, \\ A_j \cap \mathrm{post}_{\ell_j}(J_{j-1}), & j > i. \end{cases}$$

Proof. In any certificate, forward induction from (7) and (8) gives $F_j \subseteq J_j$. Backward induction from (9) gives $J_j \cap D_j = \varnothing$: a point in this intersection would have a successor in D_{j+1}, and the step condition would put that successor in J_{j+1}. This proves necessity.

For sufficiency, use the displayed sequence. The first i initial and step conditions follow from the definition of the forward sets and $F_i \subseteq K$, including the case $i = 0$. The remaining step conditions hold by construction. Disjointness from D_j propagates forward from $K \cap D_i = \varnothing$: a successor in D_{j+1} would witness that its predecessor belongs to D_j. At m, this is exactly the final condition since $J_m \subseteq A_m$ and $D_m = A_m \cap B$.

For leastness, all candidate certificates contain the exact forward sets before the pin and agree with K at the pin. Monotonicity of each forward image then propagates coordinatewise containment through the remaining positions. □

A learned predicate can be repaired by replacing K with

$$(K \cup F_i) \setminus D_i.$$

For a spurious trace, $F_i \cap D_i = \varnothing$. The formula changes only forced labels and leaves every other membership unchanged. Consequently it minimizes the weighted symmetric difference from K over all valid pins, for any nonnegative weights on A_i. With strictly positive weights it is the unique minimizer. Using exact forward or backward sets to repair a predicate can increase its representation cost.

5.1 Exclusion of all descendants

Use the whole vector to form $q'(x) = (q(x), 1_{J_0}(x), \ldots, 1_{J_m}(x))$. The descendants of an old abstract path are indexed by their refined keys. Here a descendant means one whose projections are the given old keys in the given order and whose labels agree. Different occurrences of the same old key may test different coordinates.

Theorem 5.2 (Blocking the path)**.** *If (7)–(9) hold, no bad path in the may graph of q' projects to the selected path.*

Proof. Suppose a descendant begins at key b_0. Its initial witness lies in $I \cap A_0$, so the J_0 coordinate of b_0 is true. Suppose the J_{j-1} coordinate of b_{j-1} is true. The next abstract edge has a concrete witness (x, y) with $q'(x) = b_{j-1}$, $q'(y) = b_j$. Then $x \in J_{j-1}$, $y \in A_j$, and (8) gives $y \in J_j$. Thus the J_j coordinate at b_j is true. The final bad witness would lie in $J_m \cap B$, contradiction. □

Example 5.3 (Why complete the pin)**.** Take edges $s \to u$, $u \to x$, $v \to y$, and $y \to t$, with s initial and t bad. Observe the two middle cells $\{u, v\}$ and $\{x, y\}$. At the second middle cell, $F = \{x\}$ and $D = \{y\}$. Splitting only that cell leaves the may path through y, since the earlier cell still merges u and v. The pinned completion with $K = \{x\}$ supplies the earlier predicate $\{u\}$ and blocks that descendant.

5.2 A total finite verifier

Given a finite explicitly presented graph and a seed partition, check abstract reachability. If the abstract graph is safe, project any hypothetical concrete path to get a contradiction. If an abstract bad path has a concrete lift, return a witness. Otherwise, choose a cut with both classes present when available and attempt to learn a separator. Check its completion. On failure use the exact forward sequence $J_j = F_j$. It is valid because the path was already rejected.

Each refinement strictly increases the number of realized keys. If it did not, the factor map on realized keys would be a bijection and the two may graphs would be isomorphic. The selected path would still have a descendant, contrary to Theorem 5.2. On N concrete states and k seed keys, there are at most $N - k$ refinements and $N - k + 1$ abstract checks. The result is a safety proof or a concrete bad path. This bound uses the checked elimination and holds for any pivot ordering.

6 Path fibers and internal observations

Let $\mathcal{E}$ be a Grothendieck ∞-topos. We use homotopy pullbacks, effective epimorphisms, and relative (-1)-truncations [12, 18, 15]. The boundary and edge objects of a graph may carry witnesses. A may graph replaces each image predicate by its relative support. For a generalized abstract path over U, work in the slice $\mathcal{E}/U$. All the constructions below are in that slice.

6.1 The residual at a cut

For an observed position a_i, let X_i be the homotopy fiber of the state map q over a_i. An object $L_i \to X_i$ represents concrete prefixes and their compatibility paths with the observed prefix. Dually $R_i \to X_i$ represents the suffixes. Their pullback

$$\begin{array}{ccc} L_i \times_{X_i} R_i & \longrightarrow & R_i \\ \downarrow & & \downarrow \\ L_i & \longrightarrow & X_i \end{array}$$

is the lift object of the whole path. This follows by composing the diagrams of the two path objects along their shared endpoint and using the universal property of the limit. The identification of endpoints includes their alignment with the abstract position.

Let $F_i, D_i \hookrightarrow X_i$ be the relative images of L_i, R_i. The image of the pullback above is $F_i \cap D_i \hookrightarrow X_i$. To see this, first pull back both image factorizations to the intersection. Their effective epimorphism parts remain effective epimorphisms. Their fiber product over the intersection also has this property by stability under pullback and composition. This yields the required image factorization.

Thus a refuted residual has disjoint supports in X_i. A point of the intersection expresses a local coherent lift. A global chosen lift requires a section of the pullback object itself. In the discrete finite case, these distinctions reduce to the exact state and path existence tests used above.

6.2 Internal completion

In the slice, choose an initial predicate on X_0, a final bad predicate on X_m, and transition relations between consecutive X_j, using the concrete edge objects with their endpoint alignments. Interpret forward images by $\exists$, namely relative image of the corresponding pullback. The proof of Theorem 5.1 then gives the same completion law in the lattice of subobjects. For example, if $J_{j-1} \wedge D_{j-1} = 0$ and a point of its successor image meets D_j, eliminating the existence truncation into false produces a point of $J_{j-1} \wedge D_{j-1}$. This uses no complement and no decidability of membership. The finite repair formula uses set difference and is stated for the finite Boolean setting.

For an observation on the original state object X in the slice, assume the required J_j are represented by predicates $\widehat{J}_j : X \to \Omega$ pulling back to J_j on X_j, where Ω is the subobject classifier. This holds for the discrete construction by extending each predicate outside its cell with false. The internal theorem also applies directly on a graph whose state object is the disjoint sum of the position objects X_j. On a general higher observation the extension hypothesis specifies which position predicates can be expressed on X.

Proposition 6.1 (Truth-value refinement)**.** *With the predicate extensions just specified, refine by*

$$q' = (q, \widehat{J}_0, \dots, \widehat{J}_m) : X \to A \times \Omega^{m+1}.$$

The object of refined may bad paths equipped with an identification of their projection with the selected path is initial in the slice.

Proof. Work in the context of a proposed refined descendant. The first coordinate of each key is identified with the corresponding old position. The coordinate $\widehat{J}_j$ at a key denotes a proposition. Eliminate the support of the initial witness into the first such proposition, and use the initial condition. Inductively eliminate the support of an edge witness into the next proposition. Transport across the key equalities supplies both old alignments and the previous predicate. The step condition gives the next predicate. At the final key, eliminate the support of the bad witness into false and apply the final condition. This constructs a map from the descendant object to the initial object. The initial object of an ∞-topos is strict, so the descendant object is initial. □

The proof is stable under changing the context along $V \to U$: homotopy pullback preserves the path comparison and relative images, and predicate judgments reindex. The score requires additional data, a finite weighted presentation of labeled witnesses. An equivalence preserving that labeled mass and the observations preserves the score. General context changes can change which witnesses are present and hence can change the scores assigned by an external procedure. Proposition 6.1 applies without any measure on the path fibers.

7 Effective ranking and checked completion

For a finite sample with rational weights, multiply all weights by a common denominator. The two candidate child contributions are then represented by integer radicands $a = f_0 d_0$ and $b = f_1 d_1$. The common rescaling preserves the ranking. We describe both an exact comparison used by the reference verifier and a bounded comparison checked in Lean.

7.1 An exact integer comparison

Consider two nonnegative scores $\sqrt{a} + \sqrt{b}$ and $\sqrt{c} + \sqrt{d}$. Let $p = ab$, $r = cd$, and $\delta = c + d - a - b$. Squaring both scores preserves their order. If $\delta = 0$, their order is that of p and r. If $\delta < 0$, exchange the two candidates. For $\delta > 0$, the comparison

$$\sqrt{a} + \sqrt{b} \le \sqrt{c} + \sqrt{d}$$

is equivalent to $2\sqrt{p} \le \delta + 2\sqrt{r}$. Both sides are nonnegative, so squaring again gives

$$4p - \delta^2 - 4r \le 4\delta\sqrt{r}.$$

The right side is nonnegative. If the left side is negative, the comparison is strict. If it is zero, equality holds exactly when $r = 0$. For a positive left side, comparing its square with $16\delta^2 r$ determines the order and equality. This algorithm uses only a bounded number of integer operations. It applies to ranking binary splits within the same selected cell. All unchanged cells add the same amount to each candidate.

7.2 Rational brackets with a proof

For scale $s \in \mathbb{N}_{>0}$, let

$$b_s(n) = \left\lfloor \sqrt{s^2 n} \right\rfloor .$$

A verified integer bisection returns $b = b_s(n)$ with

$$b^2 \le s^2 n < (b+1)^2.$$

Its search interval maintains a lower square at most the target and an upper square strictly above it. The midpoint replaces the appropriate endpoint. The interval length strictly decreases, and adjacent endpoints return the required bracket.

For a score pair (a, b), add the two floors. If the sum is L, its scaled true score lies in $[L, L+2)$. Choosing the least floor sum from a nonempty candidate list therefore returns a score within $2/s$ of the least true unnormalized score. For a common normalization by total witness mass $M > 0$, the error bound becomes $2/(sM)$. The Lean theorem `bestByFloor_ceiling` compares the chosen upper integer bracket with every competitor's lower bracket. The conversion of the bracket to an inequality over real square roots is the elementary argument above.

The integer products also give an exact purity test. If $f_0 d_0 = f_1 d_1 = 0$, then no positive-weight prefix and suffix witness share a branch. Lean proves this fact for any finite list of weighted witness rows. When both sides are represented, it proves that a Boolean orientation exists with all positively weighted prefix rows on one side and all positively weighted suffix rows on the other.

7.3 Lean declarations

The bundle pins Lean 4.19.0 and uses its standard library. `Entropy/Rank.lean` defines `rootFloor`, `scaledFloor_interval`, `bestByFloor_ceiling`, `zeroScore_separates`, and `zeroScore_orients`. `Entropy/Pinned.lean` defines finite forward image, backward support, and completion of a prefix layout by forward images.

The proof `completionAccepted` takes the checked logical conditions for the prefix layout through the pin and disjointness from the exact backward predicate of the remaining path. It proves that the completed layout passes the Boolean checker. `completionSound` then yields a function from any claimed bad path in that layout to false. The reusable `Cegar.Layered` module supplies reflection of the Boolean certificate checker and soundness of its result [1]. The mathematical proof of Theorem 5.2 relates that layout to exclusion in the refined may graph.

The audit rechecks the declarations and two closed examples. A four-state layout with a valid pin is accepted, and replacing that pin by its entire coarse cell is rejected. The Python program runs the complete finite refinement loop and validates each certificate and blocking claim independently. The general Arimoto calculation and infinity-topos theorems are proved in the manuscript.

8 Reproducible finite comparison

We compare four deterministic policies on the same finite graphs and seed partitions. Every policy uses breadth-first abstract reachability and exact forward and backward sets for the selected path. The first policy isolates all states in visited cells. The second refines with the exact forward predicates. The third learns a separator by the first admissible dictionary feature at each tree node. The fourth ranks those features by the exact unit-temperature score. The last two policies use the same cut rule, choosing the earliest cut that maximizes $|F_i||D_i|$ among cuts with both sides present. Both fall back to the exact forward sequence on failure.

Each run checks the certificate's initial, transition, and terminal conditions before applying it. An independent quotient-path calculation checks that the old candidate has no descendant. A separate concrete fixed-point oracle checks every final verdict. On a concrete violation the program also checks the returned path against the original boundaries, edges, and observations.

8.1 Construction of the fixtures

The 30 fixtures comprise four families. *Forks* contain two incompatible middle witness classes and a third group with no label at the cut. The dictionary orders nuisance bits before a possible class feature. Some variants omit the class feature, and some add a true violating edge. *Parity* uses individual bits to classify witnesses labeled by their parity. *Rails* has stages, cyclic lane indices, and two modes. The abstraction can fold multiple stages into the same key. A mode-changing edge introduces a real violation in some variants. *Decoys* creates multiple distinct middle cells that can each supply a spurious path. The ordering of these families and every generator parameter is included in the companion.

8.2 Outcomes

All 30 final verdicts agree with the independent oracle under every policy. Eight fixtures have a real violation. Table 1 reports sums across the common fixtures. A check counts one call to abstract reachability. The number of final cells is the realized quotient cardinality at termination, also for unsafe instances. A tree test counts an internal node in a successfully learned separator. It is a description-size measure. A fallback counts a rejected path for which the learner has no separating tree at the selected cut.

The entropy policy saves 90 internal nodes against the first-feature policy. The check and final-cell counts coincide case by case for those two policies. The difference in learned tree size comes from the fork and decoy families. They contain a feature that can separate a mixed leaf immediately and earlier nuisance features that divide the data while leaving both labels present. On the parity

Policy	Checks	Cells	Tests	Fallbacks
Bulk isolation	76	1417	–	–
Exact prefixes	81	186	–	–
First feature	83	207	181	6
Entropy rank	83	207	91	6

Table 1: Totals over 30 finite fixtures.

fixtures, both policies use 50 internal nodes in total. Every leaf of a coordinate-only parity tree must fix every bit, since varying any unfixed bit changes the label.

Exact-prefix refinement uses fewer checks and fewer final cells than either learned policy on this corpus. The chosen pin can include unlabeled states, and forward completion can propagate their additional distinctions. Bulk isolation uses the fewest checks but has a much larger final quotient. The study records separate measures for these effects. It uses deliberately structured graphs and does not report timings or statistical generalization estimates.

8.3 Two supplementary checks

The reference comparator was checked for all $12^4 = 20{,}736$ quadruples of radicands in $\{0, \dots, 11\}$ against an 80-digit decimal calculation. The proof of the comparator in Section 7 covers all natural inputs. An exhaustive regression uses every directed graph on three states, including loops, and all five partitions of those states. State zero is initial and state two is bad. Each graph/partition pair is run under all four policies, for 10,240 policy runs. Each verdict agrees with the concrete oracle, and the same certificate and descendant checks are used.

8.4 A size separation on a family

Let the concrete graph have initial state s, bad state t, and middle states $U = \{u_1, \dots, u_n\}$ and $V = \{v_1, \dots, v_n\}$. Its edges are $s \to u_j$ and $v_j \to t$. Start with one middle cell. Bulk isolation produces $2n + 2$ realized cells. A single predicate distinguishing U and V, completed as in Theorem 5.1, produces four cells. Both policies prove safety after the next check. This proves a linear quotient-size ratio on the family. Exact-prefix refinement also achieves four cells here. The comparison isolates the effect of a predicate representation from the effect of entropy as a search policy.

9 Scope and consequences

The exact tilt identity optimizes over independent probability vectors for the cells. Its predictor class expands by precomposition when a new observation coordinate is added. An additive boosted model with a coefficient budget has a different feasible set, so the entropy identity here applies directly to its partition-refit specialization. The weak-advantage bound makes explicit the condition under which a sequence of entropy pivots behaves like decision-tree boosting.

Every perfect separator ties at zero on the fully labeled cut. Entropy can guide the search while mixed leaves remain. Among completed certificates, a cost on the predicate description or on the number of realized quotient cells can resolve ties. The positional proof constraints are shared by all such choices.

For very large or symbolic systems, a candidate pool may come from interpolants, templates, or a learner using only some of the witnesses. The score can still order the proposed predicates. Exact disjointness or a proof of the logical conditions is required for acceptance. An incomplete symbolic solver can be paired with this interface and return an unresolved obligation when it lacks a certificate or a concrete witness.

The numerical optimization is performed on finite weighted evidence. The result transported through a tower of observations is a proof that the descendant object is initial. The reindexing and composition properties of that object explain why the verifier can change its pivot policy while preserving the path semantics.

References

[1] Anonymous. Counterexample-guided abstraction refinement in higher toposes. *Higher Categorical Semantics*, 11 November 2026. Editorially reviewed. Article and artifacts.

[2] Y. Ben-Haim and E. Tom-Tov. A streaming parallel decision tree algorithm. *Journal of Machine Learning Research*, 11, 849–872, 2010. Article.

[3] E. M. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith. Counterexample-guided abstraction refinement for symbolic model checking. *Journal of the ACM*, 50(5), 752–794, 2003. Author copy.

[4] M. Collins, R. E. Schapire, and Y. Singer. Logistic regression, AdaBoost and Bregman distances. *Machine Learning*, 48, 253–285, 2002. Author copy.

[5] P. Cousot and R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *POPL*, 238–252, 1977. Author copy.

[6] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1), 119–139, 1997. Article.

[7] P. Garg, D. Neider, P. Madhusudan, and D. Roth. Learning invariants using decision trees and implication counterexamples. In *POPL*, 499–512, 2016. Author copy.

[8] S. Graf and H. Saïdi. Construction of abstract state graphs with PVS. In *CAV*, 72–83, 1997. Author copy.

[9] T. A. Henzinger, R. Jhala, R. Majumdar, and G. Sutre. Lazy abstraction. In *POPL*, 58–70, 2002. Institutional record.

[10] M. Kearns and Y. Mansour. On the boosting ability of top-down decision tree learning algorithms. *Journal of Computer and System Sciences*, 58(1), 109–128, 1999. Author copy.

[11] T. Li, A. Beirami, M. Sanjabi, and V. Smith. On tilted losses in machine learning: theory and applications. *Journal of Machine Learning Research*, 24(142), 1–79, 2023. Article.

[12] J. Lurie. *Higher Topos Theory*. Annals of Mathematics Studies 170, Princeton University Press, 2009. arXiv.

[13] K. L. McMillan. Lazy abstraction with interpolants. In *CAV*, 123–136, 2006. Paper.

[14] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1, 81–106, 1986. Paper.

[15] E. Rijke, M. Shulman, and B. Spitters. Modalities in homotopy type theory. *Logical Methods in Computer Science*, 16(1), 2020. arXiv.

[16] Y. Sakai. Generalizations of Fano's inequality for conditional information measures via majorization theory. *Entropy*, 22(3), 288, 2020. Article.

[17] T. Sypherd, M. Diaz, J. K. Cava, G. Dasarathy, P. Kairouz, and L. Sankar. A tunable loss function for robust classification: calibration, landscape, and generalization. *IEEE Transactions on Information Theory*, 2022. arXiv.

[18] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. Institute for Advanced Study, 2013. Open book.