Coherent Routing in Decision Trees: Phase-Interference Learning for Interpretable Tabular Prediction

David Li
Department of Computer Science & Engineering
Yeshiva University
New York, NY, USA
david.li@yu.edu

Angela Li
Department of Computer Science
Brown University
Providence, RI, USA
angela_li4@brown.edu

Abstract

We introduce *Phase-Interference Decision Trees* (PIDT), differentiable decision trees in which each node propagates a two-dimensional, norm-preserving routing state rather than a scalar routing probability. A clipped gate splits the state's mass between the two children, a unitary branch map mixes its two components, and class scores are read from the squared component magnitudes at the leaves. The theory explains why this design is minimal: a scalar complex phase cannot change a soft decision tree, mixing is invisible unless the two components have different leaf logits, and two-state routing writes each leaf's component mass as diagonal trajectory terms plus explicit cross terms. With input-dependent branch maps the model class strictly contains soft trees, a shared two-state circuit computes parity with one module per bit, and the bounded score class admits a fixed-architecture empirical Rademacher bound. Empirically, on ten binary OpenML tasks with ten seeds each, PIDT-2S with constant rotation branch maps lowers test negative log-likelihood relative to matched scalar and soft-tree controls on all ten datasets (Wilcoxon signed-rank $p = 0.002$), by a small margin (mean 0.008), while full-size gradient-boosted ensembles remain clearly stronger. We prove that in this constant-map regime PIDT computes the same functions as a soft tree, so the measured gain reflects how the two-state readout shapes training rather than added expressivity; experiments with input-dependent branch maps, where the theory predicts a strict separation, are left open.

1 Introduction

Decision trees remain a central model class for interpretable machine learning: their branching logic is readable, their predictions are auditable, and their inference cost is small. Hard trees such as CART (Breiman et al., 1984) and C4.5 (Quinlan, 1993), and differentiable trees such as SDT (Frosst and Hinton, 2017), NODE (Popov et al., 2020) and GradTree (Marton et al., 2024a), route each input through real non-negative probabilities or hard decisions, so a differentiable tree predicts by a convex combination of leaf logits or leaf distributions. In such a model every leaf enters the score with a non-negative routing weight, so two routing paths can never cancel each other.

Complex amplitudes suggest a remedy. In physical interference, observable probabilities are squared magnitudes of complex amplitudes, and the square contains cross terms that can amplify or cancel components (Nielsen and Chuang, 2000). The obvious transfer, attaching a complex phase to every leaf, provably does nothing: Lemma 1 shows that scalar phases cancel in the squared magnitude, so a scalar-phase tree, a phase-fixed tree and a soft tree with the same gates and leaf logits form the same function class. This negative result forces the design. Cross terms require a routing state with at

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

least two components, and they reach the score only if the two components have different leaf logits (Lemma 2). PIDT is the smallest construction with both properties. At an internal node v, a clipped sigmoid gate $g_v(x) \in [\gamma, 1-\gamma]$ splits the state's mass between the children and two unitary maps mix its components:

$$a_{v_L}(x) = \sqrt{g_v(x)}\, U_{v,L}(x) a_v(x), \qquad a_{v_R}(x) = \sqrt{1-g_v(x)}\, U_{v,R}(x) a_v(x). \tag{1}$$

Here $a_v(x) \in \mathbb{C}^K$ with $K = 2$, and both children stay in $\mathbb{C}^K$. Because the branch maps are unitary and the gate factors sum to one, total leaf mass is conserved exactly. Class logits are read from the component masses $|a_{\ell,k}(x)|^2$ and passed through a softmax.

We study the model through a ladder of variants rather than a set of competing proposals. The scalar model PIDT-1S is the subject of Lemma 1 and collapses to a soft tree; the phase-fixed baseline PFB must coincide with it, which checks that the implementation matches the theory; PIDT-2S is the first rung on which the cross terms are not zero.

Contributions.

1. **Architecture.** We define PIDT as a norm-preserving two-state routing tree with explicit gate, branch-map, mass, logit and probability maps. The formulation avoids the dimensional ambiguity of using a single 2×2 matrix both as an internal-state transform and as a two-child branching map.
2. **Theory.** We prove mass conservation, scalar phase cancellation, score invisibility under tied component logits, a containment hierarchy in which two-state routing with input-dependent branch maps strictly contains soft trees, a score-level interference decomposition with explicit cross terms, a shared-circuit parity construction, and a fixed-architecture empirical Rademacher bound. We also show that with constant branch maps PIDT re-parameterises the soft-tree readout without enlarging its function class (Proposition 8).
3. **Experiments.** On ten binary OpenML tasks with ten seeds per dataset, we compare PIDT-2S with matched differentiable-tree controls using paired bootstrap intervals and dataset-level Wilcoxon tests, report CART, XGBoost and LightGBM as reference models, and measure the interference term of the trained models on the test sets. Within a fixed training budget the two-state readout lowers NLL and Brier score slightly but consistently, while boosted ensembles remain clearly stronger.

2 Related Work

Differentiable trees. Soft decision trees route with sigmoid gates and are trained by gradient descent (Frosst and Hinton, 2017). Deep neural decision forests combine such splits with random forests (Kontschieder et al., 2015), deep neural decision trees learn a soft binning of individual features (Yang et al., 2018), and adaptive neural trees grow their architecture during training (Tanno et al., 2019). NODE uses differentiable oblivious trees inside an ensemble (Popov et al., 2020), and the tree ensemble layer makes tree ensembles differentiable through conditional computation (Hazimeh et al., 2020). GradTree learns hard, axis-aligned trees by gradient descent with a straight-through estimator (Marton et al., 2024a), and GRANDE extends this to weighted tree ensembles (Marton et al., 2024b). All of these route with real non-negative weights or hard decisions. PIDT keeps a tree-indexed prediction rule and changes the routing state instead.

Tabular learning. Gradient-boosted trees such as XGBoost (Chen and Guestrin, 2016) and LightGBM (Ke et al., 2017) remain strong defaults on tabular data, as large comparative studies show (Shwartz-Ziv and Armon, 2022; Grinsztajn et al., 2022). Deep tabular models include TabNet (Arik and Pfister, 2021), FT-Transformer (Gorishniy et al., 2021), the parameter-efficient ensemble TabM (Gorishniy et al., 2025) and the pre-tuned RealMLP (Holzmüller et al., 2024). Tabular foundation models such as TabPFN (Hollmann et al., 2023, 2025; Grinsztajn et al., 2026) and TabICL (Qu et al., 2025, 2026) are pretrained on synthetic tasks and predict by in-context learning. The TabArena living benchmark (Erickson et al., 2025) compares boosted trees, deep tabular models and foundation models under a common tuning and ensembling protocol. PIDT is a single compact tree studied for its routing mechanism, not a competitor to these systems; we compare it with boosted trees (Section 6.3) but not with deep tabular or foundation models.

Table 1: Notation.

Symbol	Meaning	Symbol	Meaning
$x \in \mathbb{R}^d$, $[C]$	input, class set	$D, \mathcal{I}, \mathcal{L}_T$	depth, internal nodes, leaves
$\mathrm{path}(\ell)$, d_ℓ	root-to-leaf nodes, path length	$b_v(\ell) \in \{L, R\}$	branch taken at v
α_v, β_v, γ	gate weights, bias, clipping	$g_v(x)$	clipped gate at node v
$a_v(x) \in \mathbb{C}^K$	routing state, $a_{\mathrm{root}} = e_1$	$U_{v,b}(x)$, $R(\theta)$	branch map, plane rotation
$m_{\ell k}(x)$, $m_\ell(x)$	component and total leaf mass	$W_{\ell kc}$	component-specific leaf logits
$s_c(x)$, $p_c(x)$	class score and probability	$q_\ell(x)$	soft-tree path probability
$Q_\ell(x)$	product of branch maps on $\mathrm{path}(\ell)$	$C_{\ell k}(x)$, $I_c(x)$	cross term, interference term

Complex-valued and complex-interpreted models. Complex-valued neural networks use complex weights, activations or features for signal processing and representation learning (Trabelsi et al., 2018; Lee et al., 2022), and alternating decision trees have been given base learners that split on complex-valued features (Kuang and Ooi, 2010). Parameterised quantum circuits also compute with amplitudes and phases (Schuld et al., 2020). In PIDT the complex numbers parameterise the routing state rather than the features or the weights, and only squared magnitudes enter the score, so total routing mass is conserved exactly. In the experiments reported here the branch maps act as real rotations (Section 3.3), so no empirical claim of this paper depends on complex arithmetic.

Boolean function complexity. Parity is a standard stress test for architectures that must model high-order interactions (Linial et al., 1993; O'Donnell, 2014). Exact parity is a stringent test because it requires tracking a global interaction among all input bits. Our parity result is therefore stated only for a shared oblivious routing circuit with a two-state memory interpretation.

3 Model

3.1 Notation

Let $\mathcal{X} \subset \mathbb{R}^d$ be the input space and $[C] = \{1, \dots, C\}$ the class set. A complete binary tree $\mathcal{T}$ of depth D has internal nodes $\mathcal{I}$ and leaves $\mathcal{L}_T$, indexed in heap order so that node v has children $2v + 1$ and $2v + 2$. For a leaf ℓ, $\mathrm{path}(\ell)$ is the list of internal nodes from the root to ℓ in this order, $d_\ell = |\mathrm{path}(\ell)|$, and $b_v(\ell) \in \{L, R\}$ is the branch taken at node v; the ordering is structural and is not learned. Gates are clipped,

$$g_v(x) = \gamma + (1 - 2\gamma)\sigma(\alpha_v^\top x + \beta_v), \qquad \gamma \in (0, 1/2), \tag{2}$$

where $\alpha_v \in \mathbb{R}^d$ and $\beta_v \in \mathbb{R}$ are the gate weights and gate bias of node v and σ is the logistic sigmoid. Clipping keeps the derivatives of the square-root gate factors bounded (Section 5.2); setting $\gamma = 0$ recovers the usual sigmoid gate outside the proof assumptions. Table 1 collects the notation.

3.2 Scalar PIDT and Phase Cancellation

Definition 1 (Scalar PIDT)**.** *A scalar PIDT is a differentiable binary tree whose root amplitude is $a_{\mathrm{root}}(x) = 1 \in \mathbb{C}$. At node v, the left and right scalar amplitudes are*

$$a_{v_L}(x) = \sqrt{g_v(x)}e^{i\varphi_{v,L}(x)}a_v(x), \qquad a_{v_R}(x) = \sqrt{1 - g_v(x)}e^{i\varphi_{v,R}(x)}a_v(x), \tag{3}$$

where the phase maps are affine functions unless otherwise stated. Given leaf weights $W \in \mathbb{R}^{|\mathcal{L}_T| \times C}$, the class score and probability are

$$s_c(x) = \sum_{\ell \in \mathcal{L}_T} |a_\ell(x)|^2 W_{\ell c}, \qquad p_c(x) = \mathrm{softmax}(s(x))_c. \tag{4}$$

Lemma 1 (Scalar phase cancellation)**.** *For every scalar PIDT and every leaf ℓ,*

$$|a_\ell(x)|^2 = \prod_{v \in \mathrm{path}(\ell): b_v(\ell) = L} g_v(x) \prod_{v \in \mathrm{path}(\ell): b_v(\ell) = R} (1 - g_v(x)). \tag{5}$$

Consequently scalar PIDT, phase-fixed scalar PIDT, and a soft decision tree with the same gates and leaf logits define the same function class.

3.3 Two-State PIDT

The main PIDT model uses a K-dimensional complex state with $K = 2$ in all reported experiments. The root state is $a_{\text{root}} = e_1 \in \mathbb{C}^K$. At each internal node, the model applies the branch maps in Eq. (1), where $U_{v,L}(x), U_{v,R}(x) \in \mathcal{U}(K)$ are unitary matrices. Both child states $a_{v_L}(x)$ and $a_{v_R}(x)$ remain in $\mathbb{C}^K$; the tree branches over nodes while the latent state dimension stays fixed.

For $K = 2$, a branch map can be parameterised by real angles as

$$U_{v,b}(x) = R(\theta_{v,b}(x)) \operatorname{diag}(e^{i\rho_{v,b,1}(x)}, e^{i\rho_{v,b,2}(x)}) R(\omega_{v,b}(x)), \qquad b \in \{L, R\}, \tag{6}$$

where

$$R(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}. \tag{7}$$

The angles may be constants or affine functions of x, and the map is unitary for every input.

Implemented subfamily. All experiments in Section 6 use constant angles and the two-angle subfamily $\rho_{v,b,1} = \rho_{v,b,2} = \phi_{v,b}$, $\omega_{v,b} = 0$, that is $U_{v,b} = e^{i\phi_{v,b}} R(\theta_{v,b})$. Proposition 8 states what this choice implies: the phases $\phi_{v,b}$ cancel, the trained model acts as a real two-state rotation router, and its function class equals that of a soft tree with the same gates.

For a leaf ℓ, define component masses and total mass by

$$m_{\ell k}(x) = |a_{\ell,k}(x)|^2, \qquad m_\ell(x) = \sum_{k=1}^{K} m_{\ell k}(x) = \|a_\ell(x)\|_2^2 . \tag{8}$$

The class score and probability are

$$s_c(x) = \sum_{\ell \in \mathcal{L}_T} \sum_{k=1}^{K} m_{\ell k}(x) W_{\ell k c}, \qquad p_c(x) = \operatorname{softmax}(s(x))_c, \tag{9}$$

with $W \in \mathbb{R}^{|\mathcal{L}_T| \times K \times C}$. If all unitary maps are identity and $W_{\ell k c} = 0$ for $k > 1$, this reduces to an ordinary soft decision tree. Component-specific leaf logits are necessary for observable state-mixing effects, as formalised next.

Lemma 2 (Tied component logits make unitaries score-invisible)**.** *Suppose $W_{\ell 1 c} = \cdots = W_{\ell K c}$ for every leaf ℓ and class c. Then the score in Eq.* (9) *depends on the leaf states only through the conserved total masses $m_\ell(x)$. In particular, changing the unitary matrices while keeping the same gates cannot change the scores.*

Proof. If $W_{\ell k c} = \widetilde{W}_{\ell c}$ for all k, then

$$s_c(x) = \sum_{\ell \in \mathcal{L}_T} \widetilde{W}_{\ell c} \sum_{k=1}^{K} m_{\ell k}(x) = \sum_{\ell \in \mathcal{L}_T} \widetilde{W}_{\ell c} m_\ell(x). \tag{10}$$

For fixed gates, repeated application of Eq. (1) and unitarity gives $m_\ell(x)$ equal to the ordinary soft-tree path mass. Thus unitary rotations can redistribute mass across components but cannot change the score when component logits are tied. □

Lemma 3 (Mass conservation)**.** *For any PIDT satisfying Eq.* (1)*,*

$$\sum_{\ell \in \mathcal{L}_T} \|a_\ell(x)\|_2^2 = 1 \tag{11}$$

for every input x.

Proof. At any internal node v,

$$\|a_{v_L}(x)\|_2^2 + \|a_{v_R}(x)\|_2^2 = g_v(x) \|U_{v,L}(x) a_v(x)\|_2^2 + (1 - g_v(x)) \|U_{v,R}(x) a_v(x)\|_2^2 \tag{12}$$

$$= \|a_v(x)\|_2^2 , \tag{13}$$

since both branch maps are unitary. Induction from the root to the leaves gives the claim. □

3.4 Phase-Fixed Baseline

The phase-fixed baseline (PFB) uses the scalar model in Definition 1 with the phase parameters frozen at initialisation and the gates and leaf logits trained as usual. By Lemma 1, scalar phases do not enter the routing masses or scores once the gates and leaf logits are fixed. Under matched gate and leaf initialisation, PIDT-1S and PFB therefore implement the same effective scalar optimisation problem; the matching columns in the experimental summaries serve as a sanity check for this phase-cancellation property rather than as two independent baselines.

4 Theory

This section states the claims used by the paper with their intended scope. Full proofs and additional details are in Appendix A.

Theorem 4 (Containment hierarchy)**.** *For any fixed depth D and class count C, let the function classes below denote score-map classes, and let the overline denote closure under pointwise limits on compact input domains. The following inclusions hold up to the usual arbitrarily sharp sigmoid approximation of hard splits:*

$$\mathcal{F}_{\mathrm{hard}} \subseteq \overline{\mathcal{F}}_{\mathrm{soft}} = \overline{\mathcal{F}}_{\mathrm{PFB}} = \overline{\mathcal{F}}_{\mathrm{PIDT\text{-}1S}} \subseteq \overline{\mathcal{F}}_{\mathrm{PIDT\text{-}2S}}. \tag{14}$$

The equalities in the middle are exact before closure for the score maps in Eqs. (4) *and* (5)*. The final inclusion follows by embedding scalar routing in the first state component. The extension is observable only when component logits are not tied, as in Lemma 2. On a compact one-dimensional input domain, the final inclusion is strict already at depth one when input-dependent unitaries and component-specific leaf logits are allowed. This strictness statement is a score-level witness, not a calibrated probability-map separation.*

Lemma 5 (Score-level interference decomposition)**.** *Fix a leaf ℓ and write the product of branch unitaries along its path as $Q_\ell(x)$. For each output component k, define $\mathfrak{R}_{\ell k} = \{r = (r_0, \ldots, r_{d_\ell}) : r_0 = 1, r_{d_\ell} = k, r_j \in [K] \text{ for all } j\}$. Then*

$$a_{\ell,k}(x) = \sqrt{q_\ell(x)} \sum_{r \in \mathfrak{R}_{\ell k}} A_{\ell,k,r}(x), \tag{15}$$

where $q_\ell(x)$ is the ordinary soft-tree path probability and $A_{\ell,k,r}(x)$ is the product of unitary entries along trajectory r. Fix any ordering of the finite trajectory set and write $r < r'$ for unordered distinct trajectory pairs under that ordering. Define the real cross-term contribution

$$C_{\ell k}(x) = 2\Re \sum_{\substack{r,r' \in \mathfrak{R}_{\ell k} \\ r<r'}} A_{\ell,k,r}(x) A_{\ell,k,r'}(x)^*, \tag{16}$$

where z^* denotes complex conjugation. Taking squared modulus gives

$$m_{\ell k}(x) = q_\ell(x) \left[\sum_{r \in \mathfrak{R}_{\ell k}} |A_{\ell,k,r}(x)|^2 + C_{\ell k}(x) \right]. \tag{17}$$

The term $C_{\ell k}(x)$ is the conjugate-paired score-level cross-term contribution; it is real by construction because the real part is taken after summing unordered trajectory pairs. Therefore the PIDT score can be written as

$$s_c(x) = s_c^{\mathrm{diag}}(x) + I_c(x), \tag{18}$$

where $I_c(x)$ is the sum of all componentwise cross terms weighted by $W_{\ell k c}$. Defining $p_c^{\mathrm{diag}}(x) = \mathrm{softmax}(s^{\mathrm{diag}}(x))_c$, we have

$$p_c(x) = p_c^{\mathrm{diag}}(x) + \Delta_c(x), \qquad \sum_{c=1}^{C} \Delta_c(x) = 0. \tag{19}$$

Remark 1 (Why the decomposition is substantive)**.** *The identity $p_c = p_c^{\mathrm{diag}} + \Delta_c$ would be only notation if Δ_c were defined as the difference between two arbitrary probability vectors. Lemma 5 identifies the score-level source of the difference: the cross terms in Eq.* (17)*. These terms vanish in scalar routing and in any two-state model whose unitary products have only one nonzero latent trajectory per component; they can also be score-invisible when component logits are tied, as in Lemma 2.*

Proposition 6 (Scoped shared parity construction)**.** *Let $f_n(x) = x_1 \oplus \cdots \oplus x_n$ for $x \in \{0,1\}^n$. A shared oblivious circuit built from PIDT two-state routing modules, with a two-dimensional state, n shared modules, and a single two-component readout, computes f_n exactly. The construction uses sequential feature-indexed module sharing and is therefore outside the ordinary unshared binary-tree class.*

Remark 2 (Scope of the parity result)**.** *Proposition 6 is a circuit construction with shared modules and input-controlled branch maps. It is not a lower-bound theorem and should not be read as a claim about ordinary unshared binary trees, whose root-to-leaf paths have a different computation model. We report no parity experiment; an empirical comparison with recurrent or shared-state baselines is left to future work.*

Assumption 1 (Bounded two-state PIDT class)**.** *Inputs are augmented with a constant coordinate so that affine biases are included in the parameter vectors, and the augmented inputs satisfy $\|x\|_2 \le B_x$. Gate parameters satisfy $\|\alpha_v\|_2 \le B_g$. All angle functions in Eq.* (6) *are affine with augmented coefficient norms at most B_u. Leaf-logit tensors satisfy $\|W\|_F \le B_W$. Gates are clipped as in Eq.* (2) *with fixed $\gamma \in (0, 1/2)$.*

Theorem 7 (Fixed-architecture empirical Rademacher upper bound)**.** *Under Assumption 1, let $\mathcal{F}_{D,K}$ be the class of one-vs-all PIDT score functions $x \mapsto s_c(x)$ of depth D and state dimension K, and let $L_D \le 2^D$ be the number of leaves. For an augmented sample $S = (x_1, \ldots, x_n)$ with sample matrix X,*

$$\widehat{\mathcal{R}}_S(\mathcal{F}_{D,K}) \le \frac{C_0 B_W K L_D}{\sqrt{n}} + \frac{C_1 B_W K L_D D (B_g + B_u) \|X\|_F}{n}, \tag{20}$$

where C_0 and C_1 depend only on γ and on the fixed unitary parameterisation. Equivalently, if $n^{-1/2}\|X\|_F \le B_x$, then

$$\widehat{\mathcal{R}}_S(\mathcal{F}_{D,K}) \le \frac{B_W K L_D \left(C_0 + C_1 D (B_g + B_u) B_x\right)}{\sqrt{n}}. \tag{21}$$

Under the stated boundedness and clipping assumptions, the bound gives $n^{-1/2}$ sample-size decay with explicit depth, leaf-count, and latent-state dependence at fixed architecture size.

Remark 3 (Interpretation of the bound)**.** *Theorem 7 is stated for the two-state score class, rather than only for scalar PIDT. Its constants grow with L_D and D, so the result supports sample-size scaling at fixed architecture size. It is a capacity control on the enlarged hypothesis class: it does not quantify any benefit of interference, and we do not test it empirically.*

Proposition 8 (Constant branch maps)**.** *Suppose every branch map $U_{v,b}$ is constant in x, and let Q_ℓ be the ordered product of the branch maps along $\mathrm{path}(\ell)$. Then $m_{\ell k}(x) = q_\ell(x)\,\nu_{\ell k}$ with $\nu_{\ell k} = |(Q_\ell)_{k1}|^2$ and $\sum_k \nu_{\ell k} = 1$, so that*

$$s_c(x) = \sum_{\ell \in \mathcal{L}_T} q_\ell(x)\, \widetilde{W}_{\ell c}, \qquad \widetilde{W}_{\ell c} = \sum_{k=1}^{K} \nu_{\ell k} W_{\ell k c}. \tag{22}$$

This is the score of a soft decision tree with the same clipped gates and leaf logits $\widetilde{W}$, and every such soft-tree score is attained. PIDT-2S with constant branch maps, PIDT-1S and PFB therefore share one function class. If moreover $U_{v,b} = e^{i\phi_{v,b}} R(\theta_{v,b})$, then $\nu_{\ell 1} = \cos^2 \Theta_\ell$ and $\nu_{\ell 2} = \sin^2 \Theta_\ell$, where Θ_ℓ is the sum of the rotation angles along $\mathrm{path}(\ell)$, so no prediction depends on any phase $\phi_{v,b}$.

Remark 4 (What the experiments can and cannot test)**.** *Proposition 8 separates two regimes of the theory. The strict inclusion in Theorem 4 and the parity construction both use input-dependent branch maps. With constant maps the cross terms of Lemma 5 are real and can be large, but they only re-weight the two logit vectors of each leaf. Differences between PIDT-2S and its matched controls in this regime are therefore effects of the parameterisation on training, not of expressivity.*

5 Algorithm

5.1 Training Objective

PIDT is trained by minimising

$$\mathcal{L} = \mathcal{L}_{\mathrm{CE}} + \lambda_{\mathrm{unitary}} R_{\mathrm{unitary}} + \lambda_{\mathrm{usage}} R_{\mathrm{usage}} + \lambda_{\mathrm{floor}} R_{\mathrm{floor}}, \tag{23}$$

where $\mathcal{L}_{\mathrm{CE}}$ is cross-entropy.

Unitary smoothness regulariser. For any affine unitary angle function in Eq. (6), written generically as $\eta_{v,b,j}(x) = \eta^0_{v,b,j} + u^\top_{v,b,j} x$, we use

$$R_{\text{unitary}} = \frac{1}{|\mathcal{I}|} \sum_{v \in \mathcal{I}} \sum_{b \in \{L,R\}} \sum_{j} \|u_{v,b,j}\|_2^2 . \tag{24}$$

For constant-angle experiments, this reduces to an ℓ_2 penalty on the trainable angles.

Leaf-usage sparsity. Let $\bar{m}_\ell = \mathbb{E}_x[m_\ell(x)]$ be the mean routing mass of leaf ℓ. We use the entropy penalty

$$R_{\text{usage}} = -\sum_{\ell \in \mathcal{L}_T} \bar{m}_\ell \log(\bar{m}_\ell + 10^{-12}). \tag{25}$$

Minimising this term favours concentrated leaf usage.

Leaf-floor regulariser. A pure entropy penalty can assign nearly all mass to a very small number of leaves. We therefore add

$$R_{\text{floor}} = \log \sum_{\ell \in \mathcal{L}_T} \exp(-\bar{m}_\ell / \tau), \qquad \tau = 0.1, \tag{26}$$

which discourages configurations with too many near-zero leaf masses. The two usage terms are deliberately in tension: R_{usage} favours a compact routing support, while R_{floor} keeps multiple leaves active.

5.2 Optimisation

The square-root gates $\sqrt{g_v(x)}$ in Eqs. (1) and (3) are differentiable, but $\partial\sqrt{g}/\partial g = 1/(2\sqrt{g})$ is unbounded as $g \to 0$. The clipped gate of Eq. (2) bounds this derivative by $1/(2\sqrt{\gamma})$ and the derivative with respect to the gate pre-activation by $(1-2\gamma)/(8\sqrt{\gamma})$. We fix $\gamma = 10^{-2}$, which gives the bounds 5 and 1.23, for every model, dataset and seed. It is a numerical safeguard chosen for this analytic reason rather than a tuned hyperparameter, and we did not sweep it. We additionally clip the global L_2 gradient norm to $G_{\max} = 1$ so that the branch angles cannot make arbitrarily large jumps early in training.

Proposition 9 (Clipped raw angle gradient)**.** *If the unitary-angle parameter group is clipped so that its Euclidean gradient norm is at most $G_{\max}$ before an optimiser step, then the raw gradient vector passed to the optimiser has norm at most $G_{\max}$. For plain gradient descent with learning rate η, the corresponding per-step displacement is at most $\eta G_{\max}$, and the raw path length after T steps is at most $\eta G_{\max} T$. For Adam, the final adaptive displacement also depends on the optimiser state, so the proposition is used only as a gradient-stability diagnostic.*

Computational cost. A differentiable tree evaluates every internal node for every example, whereas a hard tree follows one path. Per example, an SDT forward pass costs $O(2^D(d+C))$ for the gates and the leaf readout; PIDT-2S adds one $K \times K$ state map per node and K readouts per leaf, giving $O(2^D(d+K^2+KC))$, the same order for $K = 2$. With constant branch maps the maps are shared across a batch. A hard tree of the same depth costs $O(D)$ at inference. Measured training wall clock per run at depth six on a CPU, averaged over the ten datasets, is 84 s for PIDT-2S, 42 s for PIDT-1S, 30 s for PFB and 14 s for SDT, against under 1 s for the CART, XGBoost and LightGBM references (Appendix C.5). PIDT-2S trains 5.4 to 6.5 times slower than SDT across datasets. We did not profile inference separately.

Algorithm 1 summarises the training loop used for PIDT-2S.

6 Experiments

We ask whether PIDT-2S changes NLL and Brier score reliably relative to matched differentiable-tree controls, where it stands among standard tree models, and whether the trained model uses the cross terms of Lemma 5.

Algorithm 1 PIDT-2S training

Require: Training set $\{(x_i, y_i)\}_{i=1}^{n}$, depth D, $K = 2$, weights $\lambda_{\text{unitary}}, \lambda_{\text{usage}}, \lambda_{\text{floor}}$, learning rate η, clipping norm $G_{\max}$, gate clip γ, angle scale s

Draw gate weights and the leaf logits $W_{\ell,k,c}$ of both components from $\mathcal{N}(0, 0.01)$
Draw the angles $\theta_{v,b}, \phi_{v,b} \sim U(-\pi s/4, \pi s/4)$ of $U_{v,b} = e^{i\phi_{v,b}} R(\theta_{v,b})$ (default $s = 1$)
for each epoch **do**
    for each mini-batch $\mathcal{B}$ of the shuffled training set **do**
        Compute gates (Eq. (2)), leaf states (Eq. (1)), probabilities (Eq. (9)) and the loss (Eq. (23))
        Backpropagate, clip the global gradient norm to $G_{\max}$, and take an Adam step
    end for
    Keep the parameters with the best validation NLL; stop after T_{patience} epochs without improvement
end for
return the best-validation model

6.1 Setup

Datasets. We use ten binary OpenML tasks (Vanschoren et al., 2014) (Table 6 in Appendix C). Datasets with more than $10{,}000$ examples are subsampled to $10{,}000$ per seed before splitting. For every dataset and seed we draw a stratified 80/20 train-test split and hold out 15% of the training split for validation. Numeric features use median imputation and standard scaling; categorical features use most-frequent imputation and ordinal encoding with unknown categories mapped to -1. All preprocessing is fit on the training split only.

Models. The matched controls are PIDT-1S (Definition 1), PFB, which freezes the PIDT-1S phases at initialisation, and SDT (Frosst and Hinton, 2017), all at depth $D = 6$. As reference models we report CART with 64 leaves, matching a depth-six tree, with the minimum leaf size chosen on validation NLL (CART-64); a CART tuned over depth $\{6, 8, 10, 12\}$ and minimum leaf size $\{1, 5, 10\}$ on validation NLL (CART-tuned); and XGBoost (Chen and Guestrin, 2016) and LightGBM (Ke et al., 2017) in a small configuration (64 trees, depth 3 or 31 leaves; XGB-64, LGBM-64) and a full one (300 trees, depth 6; XGB-300, LGBM-300), all with early stopping on validation log-loss. CART uses scikit-learn (Pedregosa et al., 2011). The reference models are neither capacity matched nor tuned beyond these grids.

Training protocol. All differentiable models use batch size 256, Adam with learning rate 10^{-3} and weight decay 10^{-5}, gradient-norm clipping at 1, early stopping on validation NLL with patience 20, and a 100-epoch budget. PIDT-2S uses $\gamma = 10^{-2}$, $\lambda_{\text{unitary}} = \lambda_{\text{usage}} = \lambda_{\text{floor}} = 10^{-3}$, branch angles $\theta_{v,b}, \phi_{v,b} \sim U(-\pi/4, \pi/4)$ at initialisation and leaf logits $W_{\ell,k,c} \sim \mathcal{N}(0, 0.01)$ for both components. The usage and floor regularisers apply to all four differentiable models and the angle penalty to the models with trainable angles or phases. Every model is trained with seeds $\{0, \ldots, 9\}$ on every dataset, and all models share each seed's split.[1] The budget binds: in 387 of the 400 differentiable-tree runs the best validation NLL falls in the last two epochs, so the comparison is matched at a fixed training budget rather than at convergence.

Metrics and statistics. We report accuracy, NLL, Brier score and ECE with 15 bins (Guo et al., 2017). Paired comparisons use the same test examples for both models in each seed. For each dataset and baseline, a paired bootstrap with 2000 resamples of the test set per seed gives a 95% interval for the seed-averaged difference; a dataset is a significant win (loss) when the interval lies above (below) zero. Across datasets we use two-sided exact Wilcoxon signed-rank tests on the ten per-dataset mean differences, with the dataset as the unit of analysis (Wilcoxon, 1945; Demšar, 2006). Differences are baseline minus PIDT-2S for NLL, Brier score and ECE and the reverse for accuracy, so positive values favour PIDT-2S. Holm-adjusted values over all 18 dataset-level tests are in Appendix C.2.

Scope. The experiments test only constant branch maps of the form $e^{i\phi} R(\theta)$. By Proposition 8, PIDT-2S, PIDT-1S and PFB then share one function class (SDT differs only in using unclipped

[1] The submitted version of this paper used five seeds on seven of the ten datasets. All results here come from the completed ten-seed sweep; no conclusion changed.

Table 2: PIDT-2S against the matched controls over ten datasets and ten seeds. Δ is the mean over datasets of the per-dataset paired difference (baseline minus PIDT-2S, so positive favours PIDT-2S); "pos." counts datasets with a positive difference; "boot." gives significant bootstrap wins and losses; p is the two-sided Wilcoxon signed-rank test over datasets. PIDT-1S and PFB coincide, as Lemma 1 predicts.

	NLL				Brier score			
Control	Δ	pos.	boot.	p	Δ	pos.	boot.	p
PIDT-1S / PFB	$+0.0083$	10/10	9/0	0.002	$+0.0028$	10/10	9/0	0.002
SDT	$+0.0084$	10/10	9/0	0.002	$+0.0028$	10/10	6/0	0.002

Table 3: PIDT-2S against reference tree models: significant bootstrap wins and losses over ten datasets, and the two-sided Wilcoxon p for NLL. The reference models are not capacity matched to PIDT-2S.

Model	Accuracy	NLL	Brier	ECE	p (NLL)
CART-64	3/4	9/0	3/4	4/3	0.010
CART-tuned	3/2	9/0	3/3	1/5	0.004
XGB-64	3/4	2/7	2/7	7/2	0.131
XGB-300	0/10	0/10	0/10	0/8	0.002
LGBM-64	0/9	0/10	0/10	3/3	0.002
LGBM-300	0/10	0/10	0/10	0/7	0.002

gates), so differences between them are effects of how the readout is parameterised during training. Input-dependent branch maps are not evaluated, and we report no parity experiment.

6.2 Matched-Control Results

Table 2 summarises the matched comparison. PIDT-2S has lower NLL than PIDT-1S, PFB and SDT on all ten datasets ($p = 0.002$, the smallest value a two-sided exact test on ten datasets can give), with significant bootstrap wins on nine and no significant loss, and lower Brier score on all ten. The effect is small: the mean differences are 0.008 in NLL and 0.003 in Brier score. Accuracy differences are mostly not significant (two or three significant wins and no losses per control). PIDT-1S and PFB give identical results, as Lemma 1 requires. Per-dataset values for all ten models and four metrics are in Appendix C.3. On Banknote and QSAR Biodeg, two of the three smallest datasets, all four differentiable trees are poorly calibrated relative to every tree model; on Banknote their ECE is 0.28 to 0.29 against at most 0.06 (Table 12). This affects PIDT-2S and its controls alike and is consistent with the binding training budget.

6.3 Reference Tree Models

Table 3 places PIDT-2S among standard tree models. Full-size boosted ensembles beat it significantly on accuracy, NLL and Brier score on every dataset, and LightGBM with 64 trees does so on NLL and Brier score on every dataset and on accuracy on nine. Against XGBoost with 64 trees, the NLL and Brier differences are not significant across datasets ($p = 0.13$ and $p = 0.11$) and PIDT-2S has significantly lower ECE on seven datasets and higher on two. Against CART, PIDT-2S has significantly lower NLL on nine of ten datasets, while accuracy and Brier score are mixed, the expected profile of a smooth probabilistic router compared with a hard partition. PIDT-2S is therefore not a competitive tabular predictor in this regime.

6.4 Mechanism Diagnostics

Interference on the test sets. For constant branch maps the cross term of Lemma 5 can be computed exactly. With P_ℓ the product of the entrywise squared magnitudes of the branch maps along $\mathrm{path}(\ell)$, the trajectory sum is $(P_\ell)_{k1}$ and $C_{\ell k} = |(Q_\ell)_{k1}|^2 - (P_\ell)_{k1}$. Because P_ℓ is doubly stochastic and Q_ℓ

is unitary, $\sum_k C_{\ell k} = 0$, and for $K = 2$

$$I_c(x) = \sum_{\ell \in \mathcal{L}_T} q_\ell(x)\, C_{\ell 1}\, (W_{\ell 1 c} - W_{\ell 2 c}), \tag{27}$$

which makes Lemma 2 quantitative: tying the logits sets I_c to zero. Over the 100 trained models, on their test splits, the cross terms are substantial (Table 4 in Appendix B). Their size, $\max_c |I_c(x)|$, is on average 0.48 of the spread of the class scores (0.23 to 1.68 by dataset), and removing them at inference changes 1.5% of predicted labels and raises mean test NLL from 0.365 to 0.424, with an increase on every dataset. This deletion is applied to an already trained model. It measures how the trained readout distributes its weight between the diagonal and cross terms, not what a model trained without cross terms would reach; by Proposition 8 such a model could represent the same functions.

Phases and the soft-tree form. Setting every phase $\phi_{v,b}$ to zero, or redrawing all of them uniformly on $[-\pi, \pi]$, changes no test prediction by more than 4.8×10^{-7}, while shifting all left-branch rotation angles by 0.05 changes predictions by 1.8×10^{-3} to 5.0×10^{-2}, so the test can detect real changes. Rewriting each trained model as the soft tree of Eq. (22) reproduces its predictions to within 4.1×10^{-7}. Both results confirm Proposition 8 numerically.

Controls on the source of the effect. Tying the two component logits keeps the architecture, parameter count and training protocol fixed and, by Lemma 2, makes the rotations invisible in the score. On the three ablation datasets, three seeds each, test NLL rises from 0.3449 to 0.3555 on Adult, from 0.3395 to 0.3421 on MagicTelescope and from 0.4316 to 0.4596 on Banknote (Table 13). Raw parameter counts settle nothing here: PIDT-2S has fewer nominal parameters than PIDT-1S (mean 1862 against 4192), but the PIDT-1S phases are inert by Lemma 1, leaving it the size of SDT (mean 1482). The regularisers do not explain the effect either: a single-axis sweep of each weight over $\{0, 10^{-4}, 10^{-3}, 10^{-2}\}$ keeps mean accuracy between 87.88% and 88.32% and mean NLL between 0.3704 and 0.3834 (Table 14).

7 Conclusion

PIDT recovers soft trees in the scalar case and exposes explicit score-level cross terms when the component logits differ. With input-dependent branch maps it strictly extends soft trees and computes parity with a linear-size shared circuit; with constant maps it re-parameterises the soft-tree readout. In that regime the two-state readout lowers NLL and Brier score consistently but slightly within a fixed training budget, the trained models use their cross terms substantially, and boosted ensembles remain clearly stronger. The contribution is the mechanism and its theory, not a new level of tabular performance.

Limitations. The experiments use constant branch maps of the form $e^{i\phi}R(\theta)$, so they do not exercise input-dependent maps, the general four-angle family of Eq. (6), or the regime in which the theory predicts a strict separation from soft trees. All differentiable models are trained under a fixed budget that binds, and the source of the PIDT-2S advantage within it, for example initialisation or the implicit regularisation of the two-state readout, is not isolated. All tasks are binary, datasets above $10{,}000$ examples are subsampled, and there is no evidence for multi-class problems, larger data, deeper trees or high-dimensional inputs. Depth, state dimension, γ and the angle initialisation were held fixed and not swept. We did not compare with deep tabular models or tabular foundation models, nor with recurrent or shared-state baselines on parity. Forests of PIDT trees, including interference between trees through a shared readout, are left to future work.

Broader impact. PIDT carries the standard risks of classifiers (dataset bias, calibration error, misread explanations), makes no deployment claim and adds no new misuse channel; exact mass conservation and the interference decomposition make its routing easier to inspect.

Acknowledgments and Disclosure of Funding

We thank the anonymous reviewers and the area chair for detailed reviews and discussion that improved this paper. This work was supported by Yeshiva University, which funded conference registration and related expenses. The authors have no competing interests to declare.

References

Sercan Ö. Arik and Tomas Pfister. TabNet: Attentive interpretable tabular learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(8):6679–6687, 2021.

Peter L. Bartlett and Shahar Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. *Classification and Regression Trees*. Wadsworth, 1984.

Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, 2016.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

Nick Erickson, Lennart Purucker, Andrej Tschalzev, David Holzmüller, Prateek Desai, David Salinas, and Frank Hutter. TabArena: A living benchmark for machine learning on tabular data. In *Advances in Neural Information Processing Systems 38, Datasets and Benchmarks Track*, pages 17285–17350, 2025.

Nicholas Frosst and Geoffrey Hinton. Distilling a neural network into a soft decision tree. *arXiv preprint arXiv:1711.09784*, 2017.

Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. In *Advances in Neural Information Processing Systems 34*, pages 18932–18943, 2021.

Yury Gorishniy, Akim Kotelnikov, and Artem Babenko. TabM: Advancing tabular deep learning with parameter-efficient ensembling. In *International Conference on Learning Representations (ICLR)*, 2025.

Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. Why do tree-based models still outperform deep learning on typical tabular data? In *Advances in Neural Information Processing Systems 35, Datasets and Benchmarks Track*, pages 507–520, 2022.

Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, Mihir Manium, Shi Bin Hoo, Magnus Bühler, Anurag Garg, Dominik Safaric, Jake Robertson, Benjamin Jäger, Simone Alessi, Adrian Hayler, Vladyslav Moroshan, Lennart Purucker, Philipp Singer, Alan Arazi, Julien Siems, Jan Hendrik Metzen, Georg Grab, Nick Erickson, Siyuan Guo, Eliott Kalfon, Simon Bing, David Salinas, Clara Cornu, Lilly Charlotte Wehrhahn, Diana Kriuchkova, Kursat Kaya, Lydia Sidhoum, Marie Salmon, Jerry Chen, Madelon Hulsebos, Yann LeCun, Samuel Müller, Bernhard Schölkopf, Sauraj Gambhir, Noah Hollmann, and Frank Hutter. TabPFN-3: Technical report. *arXiv preprint arXiv:2605.13986*, 2026.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330, 2017.

Hussein Hazimeh, Natalia Ponomareva, Petros Mol, Zhenyu Tan, and Rahul Mazumder. The tree ensemble layer: Differentiability meets conditional computation. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4138–4148, 2020.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *International Conference on Learning Representations (ICLR)*, 2023.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637(8045):319–326, 2025.

David Holzmüller, Léo Grinsztajn, and Ingo Steinwart. Better by default: Strong pre-tuned MLPs and boosted trees on tabular data. In *Advances in Neural Information Processing Systems 37*, pages 26577–26658, 2024.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems 30*, pages 3146–3154, 2017.

Peter Kontschieder, Madalina Fiterau, Antonio Criminisi, and Samuel Rota Bulò. Deep neural decision forests. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1467–1475, 2015.

Ye Chow Kuang and Melanie Po-Leen Ooi. Complex feature alternating decision tree. *International Journal of Intelligent Systems Technologies and Applications*, 9(3/4):335–353, 2010.

ChiYan Lee, Hideyuki Hasegawa, and Shangce Gao. Complex-valued neural networks: A comprehensive survey. *IEEE/CAA Journal of Automatica Sinica*, 9(8):1406–1426, 2022.

Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, Fourier transform, and learnability. *Journal of the ACM*, 40(3):607–620, 1993.

Sascha Marton, Stefan Lüdtke, Christian Bartelt, and Heiner Stuckenschmidt. GradTree: Learning axis-aligned decision trees with gradient descent. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(13):14323–14331, 2024.

Sascha Marton, Stefan Lüdtke, Christian Bartelt, and Heiner Stuckenschmidt. GRANDE: Gradient-based decision tree ensembles for tabular data. In *International Conference on Learning Representations (ICLR)*, 2024.

Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Sergei Popov, Stanislav Morozov, and Artem Babenko. Neural oblivious decision ensembles for deep learning on tabular data. In *International Conference on Learning Representations (ICLR)*, 2020.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICL: A tabular foundation model for in-context learning on large data. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 50817–50847, 2025.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICLv2: A better, faster, scalable, and open tabular foundation model. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 102477–102555, 2026.

J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

Maria Schuld, Alex Bocharov, Krysta M. Svore, and Nathan Wiebe. Circuit-centric quantum classifiers. *Physical Review A*, 101(3):032308, 2020.

Ravid Shwartz-Ziv and Amitai Armon. Tabular data: Deep learning is not all you need. *Information Fusion*, 81:84–90, 2022.

Ryutaro Tanno, Kai Arulkumaran, Daniel Alexander, Antonio Criminisi, and Aditya Nori. Adaptive neural trees. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6166–6175, 2019.

Chiheb Trabelsi, Olexa Bilaniuk, Ying Zhang, Dmitriy Serdyuk, Sandeep Subramanian, João Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Yoshua Bengio, and Christopher J. Pal. Deep complex networks. In *International Conference on Learning Representations (ICLR)*, 2018.

Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. OpenML: Networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 15(2):49–60, 2014.

Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945.

Yongxin Yang, Irene Garcia Morillo, and Timothy M. Hospedales. Deep neural decision trees. In *ICML 2018 Workshop on Human Interpretability in Machine Learning (WHI)*, 2018. *arXiv preprint arXiv:1806.06988*.

A Proofs

A.1 Proof of Lemma 1

For a scalar path, multiplying Eq. (3) along the path to leaf ℓ gives

$$a_\ell(x) = \prod_{v \in \mathrm{path}(\ell): b_v(\ell)=L} \sqrt{g_v(x)} e^{i\varphi_{v,L}(x)} \prod_{v \in \mathrm{path}(\ell): b_v(\ell)=R} \sqrt{1-g_v(x)} e^{i\varphi_{v,R}(x)}. \tag{28}$$

Taking squared modulus removes every unit complex phase and gives Eq. (5). Substituting these masses into Eq. (4) gives exactly the soft-tree score with the same leaf logits.

A.2 Proof of Theorem 4

A hard split can be approximated by a sigmoid split $\sigma(t(\alpha_v^\top x + \beta_v))$ as t grows, which gives the first inclusion in the closure sense. Lemma 1 proves $\mathcal{F}_{\mathrm{soft}} = \mathcal{F}_{\mathrm{PFB}} = \mathcal{F}_{\text{PIDT-1S}}$ for the score maps before closure. The inclusion into $\mathcal{F}_{\text{PIDT-2S}}$ follows by setting all unitary maps to identity, initialising the root state as e_1, and setting $W_{\ell kc} = 0$ for every $k > 1$.

For the stated strictness witness at depth one, fix $g_v(x) = 1/2$ on $x \in [-\pi, \pi]$ and use an input-dependent rotation with angle $\theta(x) = x$ on one branch. With component-specific logits, a two-state score can contain a nonconstant term proportional to $\cos^2 x$. A depth-one scalar soft tree with an affine sigmoid gate has scores of the form $a + b\sigma(cx + d)$, which are monotone when $bc \neq 0$ and constant otherwise. It cannot equal the nonmonotone function $\cos^2 x$ on $[-\pi, \pi]$. Lemma 2 explains why component-specific logits are needed for this witness. This proves strictness for the scoped depth-one claim.

A.3 Proof of Lemma 5

Fix a leaf ℓ. Along the path to ℓ, the scalar gate factors separate from the unitary product:

$$a_\ell(x) = \sqrt{q_\ell(x)} Q_\ell(x) e_1, \tag{29}$$

where

$$q_\ell(x) = \prod_{v: b_v(\ell)=L} g_v(x) \prod_{v: b_v(\ell)=R} (1 - g_v(x)) \tag{30}$$

and $Q_\ell(x)$ is the ordered product of branch unitaries. Expanding the matrix product entry $(Q_\ell)_{k1}$ as a sum over the trajectory set $\mathfrak{R}_{\ell k}$ yields Eq. (15). Expanding the squared modulus gives the diagonal term and the real conjugate-paired component term $C_{\ell k}(x)$ in Eq. (16); substituting this into the component mass gives Eq. (17). Substitution into Eq. (9) separates diagonal terms from cross terms, giving Eq. (18). Applying softmax gives Eq. (19), and the sum-to-zero property of Δ follows because both p and p^{diag} are probability vectors.

A.4 Proof of Proposition 6

Use a two-state vector whose basis states represent even and odd parity. Start from $e_0 = (1, 0)^\top$. For bit x_j, apply the input-controlled unitary

$$U_j(x_j) = \begin{cases} I, & x_j = 0, \\ X, & x_j = 1, \end{cases} \qquad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}. \tag{31}$$

After applying $U_n(x_n) \cdots U_1(x_1)$, the state is e_0 if the number of ones is even and e_1 if the number of ones is odd. A linear readout that maps e_0 to class 0 and e_1 to class 1 computes parity exactly. The construction uses one shared module per bit and a single two-component readout. It is a shared oblivious circuit because the modules are arranged by feature index rather than by the nodes of a full unshared binary tree.

Lemma 10 (Bounded and Lipschitz component masses)**.** *Under Assumption 1, each component mass $m_{\ell k} : x \mapsto |a_{\ell,k}(x)|^2$ is bounded in $[0, 1]$. Moreover, for the fixed unitary parameterisation in Eq.* (6)*, there is a constant C_m depending only on γ and the parameterisation such that every $m_{\ell k}$ is $C_m D(B_g + B_u)$-Lipschitz on the augmented input domain.*

Proof. Boundedness follows from mass conservation: each leaf-state norm is at most one, so every component mass is in $[0, 1]$. For Lipschitzness, the clipped square-root gate factors are Lipschitz with constant controlled by γ and B_g, while each unitary entry is a smooth function of affine angles whose coefficient norms are bounded by B_u. Along any root-to-leaf path of length at most D, products of bounded factors are therefore Lipschitz with constant at most $CD(B_g + B_u)$ for a constant depending only on γ and the unitary parameterisation. Since $z \mapsto |z|^2$ is 2-Lipschitz on the unit disk, the same order bound holds for $m_{\ell k}$ after adjusting the constant. □

A.5 Proof of Theorem 7

By Lemma 10, each component-mass function is bounded in $[0, 1]$ and has Lipschitz constant at most $C_m D(B_g + B_u)$. For a fixed class c, the score is a sum of at most KL_D component-mass functions weighted by $W_{\ell kc}$. The bounded component-mass part contributes the first $O(B_W KL_D/\sqrt{n})$ term in Eq. (20). The Lipschitz part contributes the second term by Cauchy-Schwarz and the contraction lemma for Rademacher averages (Bartlett and Mendelson, 2002), with the factor $\|X\|_F / n$ coming from the sample-dependent Lipschitz contraction. Combining these two pieces gives Eq. (20). If $n^{-1/2} \|X\|_F \le B_x$, this becomes Eq. (21). Standard multiclass bounds then apply to softmax cross-entropy through the usual Lipschitz contraction argument, with an additional constant depending on the chosen multiclass loss. The theorem is used only as a fixed-architecture capacity control, not as a claim that deeper trees have depth-independent complexity.

A.6 Proof of Proposition 8

With constant branch maps, Q_ℓ does not depend on x, and the factorisation in the proof of Lemma 5 gives $a_\ell(x) = \sqrt{q_\ell(x)}\, Q_\ell e_1$. Hence $m_{\ell k}(x) = q_\ell(x)|(Q_\ell)_{k1}|^2 = q_\ell(x)\nu_{\ell k}$, and unitarity gives $\sum_k \nu_{\ell k} = \|Q_\ell e_1\|_2^2 = 1$. Substituting into Eq. (9) gives Eq. (22). Identity branch maps give $\nu_{\ell 1} = 1$, so choosing $W_{\ell 1 c} = \widetilde{W}_{\ell c}$ attains every soft-tree score with the same gates. By Lemma 1 the scores of PIDT-1S and PFB are exactly these soft-tree scores, so the three models share one function class. If $U_{v,b} = e^{i\phi_{v,b}} R(\theta_{v,b})$, the scalar factors commute with every rotation and $R(\theta)R(\theta') = R(\theta + \theta')$, so $Q_\ell = e^{i\Phi_\ell} R(\Theta_\ell)$, where Φ_ℓ and Θ_ℓ are the sums of the phases and rotation angles along $\mathrm{path}(\ell)$. Then $(Q_\ell)_{11} = e^{i\Phi_\ell} \cos\Theta_\ell$ and $(Q_\ell)_{21} = e^{i\Phi_\ell} \sin\Theta_\ell$, so $\nu_{\ell 1} = \cos^2\Theta_\ell$, $\nu_{\ell 2} = \sin^2\Theta_\ell$, and the phases drop out.

B Mechanism Diagnostics

B.1 Computing the Interference Term

For each trained PIDT-2S model and leaf ℓ, we form the ordered product Q_ℓ of the branch maps along $\mathrm{path}(\ell)$ and the product P_ℓ of their entrywise squared magnitudes. Expanding $(Q_\ell)_{k1}$ over latent trajectories as in Lemma 5 shows that $\sum_r |A_{\ell,k,r}|^2 = (P_\ell)_{k1}$, because each trajectory contributes the product of the squared magnitudes of its matrix entries. The cross term is then $C_{\ell k} = |(Q_\ell)_{k1}|^2 - (P_\ell)_{k1}$, which with constant maps does not depend on x. Each factor of P_ℓ is doubly stochastic because the branch maps are unitary, so $\sum_k (P_\ell)_{k1} = 1$, and $\sum_k |(Q_\ell)_{k1}|^2 = 1$, so $\sum_k C_{\ell k} = 0$ and Eq. (27) follows. On every test example we compute $s_c(x)$, the diagonal score $s_c^{\mathrm{diag}}(x) = s_c(x) - I_c(x)$ and the softmax of each. Table 4 reports $\max_c |I_c(x)|/(\max_c s_c(x) - \min_c s_c(x))$ averaged over test examples, the fraction of test examples whose $\arg\max$ differs between p and p^{diag}, and the two test NLLs; Table 5 gives the values per dataset. Each value is averaged over the ten seeds. The submitted version of this paper reported only an identity-routing residual, which replaces every branch map by the identity; that residual is a proxy and is superseded by the exact term above.

C Reproduction Protocol and Additional Results

C.1 Data and Commands

Table 6 lists the datasets. The commands below reproduce every table in the paper. The flags `--lambda_phase`, `--lambda_sparsity` and `--lambda_collapse` are the implementation names of $\lambda_{\mathrm{unitary}}$, λ_{usage} and λ_{floor} in Eq. (23).

Table 4: Mechanism diagnostics for the trained PIDT-2S models, computed on the test split of every dataset and seed (100 checkpoints).

Diagnostic	Result
Interference size, $\max_c \lvert I_c(x)\rvert$ relative to the class-score spread	0.48 (0.23 to 1.68 by dataset)
Test points whose predicted label changes when I_c is removed	1.5%
Test NLL with and without the cross terms	0.365 and 0.424; higher on 10/10
Largest prediction change, all phases set to zero or redrawn	$\leq 4.8 \times 10^{-7}$
Positive control: all left-branch angles shifted by 0.05	1.8×10^{-3} to 5.0×10^{-2}
Largest gap to the soft-tree form of Eq. (22)	$\leq 4.1 \times 10^{-7}$

Table 5: Interference diagnostics per dataset, mean over ten seeds. "Size" is $\max_c \lvert I_c(x)\rvert$ relative to the class-score spread; "flips" is the percentage of test points whose predicted label changes when I_c is removed.

Dataset	Size	Flips (%)	NLL	NLL without I_c	Difference
Adult	0.438	2.49	0.3484	0.4132	+0.0648
Bank Marketing	0.293	0.87	0.2467	0.3082	+0.0615
Electricity	0.360	3.02	0.4646	0.5026	+0.0380
MagicTelescope	0.632	1.94	0.3381	0.3925	+0.0544
Phishing	1.679	0.69	0.1724	0.2336	+0.0612
Spambase	0.229	0.28	0.2036	0.2841	+0.0805
Banknote	0.239	0.40	0.4305	0.5097	+0.0792
Credit-g	0.273	1.10	0.5733	0.6049	+0.0316
Phoneme	0.382	2.89	0.4090	0.4594	+0.0504
QSAR Biodeg	0.249	1.18	0.4656	0.5317	+0.0662

```
# 1. Environment (Python 3.12, CPU)
pip install -r requirements.txt
pip install torch --index-url https://download.pytorch.org/whl/cpu

# 2. Data: cache the ten OpenML datasets
python -m experiments_lowclass.suite

# 3. Main sweep: 10 models x 10 datasets x 10 seeds
python -m experiments_lowclass.run_suite --tag pidt_lowclass_10 \
  --seeds 0 1 2 3 4 5 6 7 8 9 --depth 6 --epochs 100 \
  --batch_size 256 --lr 1e-3 --weight_decay 1e-5 \
  --lambda_phase 1e-3 --lambda_sparsity 1e-3 --lambda_collapse 1e-3 \
  --phase_clip 1.0 --patience 20 --max_samples 10000

# 4. Tied-logit control and regulariser sweep (3 datasets x 3 seeds)
python -m experiments_lowclass.run_ablations --mode ablation
python -m experiments_lowclass.run_ablations --mode lambda_grid \
  --tag pidt_lowclass_lambda_grid

# 5. Aggregation, paired bootstrap, tests, diagnostics and tables
python -m experiments_lowclass.aggregate --tag pidt_lowclass_10 \
  --n_boot 2000 --ci 0.95
python -m analysis.mechanism
python -m analysis.tables
python -m analysis.training_budget
```

C.2 Statistical Tests

Table 7 reports the dataset-level Wilcoxon signed-rank tests for all nine baselines. Each test is two-sided and exact, on the ten per-dataset mean paired differences; with ten datasets and all differences of one sign the smallest attainable value is $2/2^{10} \approx 0.002$. Holm's procedure is applied across all 18

Table 6: The ten OpenML datasets. "Instances" is the per-seed sample size after the $10{,}000$-example cap; "Pos." is the positive-class percentage.

Dataset	OpenML ID	Instances	Features	Pos. (%)
Adult Income	179	10000	14	23.7
Bank Marketing	1461	10000	16	11.4
Electricity	151	10000	8	42.4
MagicTelescope	1120	10000	10	35.5
PhishingWebsites	4534	10000	30	55.6
Spambase	44	4601	57	39.4
Banknote Authentication	1462	1372	4	44.5
Credit-g	31	1000	20	70.0
Phoneme	1489	5404	5	29.3
QSAR Biodeg	1494	1055	41	33.7

tests. The bootstrap intervals behind the win and loss counts in Tables 2 and 3 keep the test-example pairing within each seed, resample the test set 2000 times per seed, and average the resampled differences over seeds.

Table 7: Dataset-level Wilcoxon signed-rank tests, PIDT-2S against each baseline. "Median" is the median over datasets of the per-dataset paired difference (baseline minus PIDT-2S); p_{Holm} is adjusted over all 18 tests.

	NLL			Brier score		
Baseline	Median	p	p_{Holm}	Median	p	p_{Holm}
PIDT-1S	$+0.0048$	0.002	0.035	$+0.0013$	0.002	0.035
PFB	$+0.0048$	0.002	0.035	$+0.0013$	0.002	0.035
SDT	$+0.0073$	0.002	0.035	$+0.0017$	0.002	0.035
CART-64	$+0.1322$	0.010	0.049	-0.0013	0.625	1.000
CART-tuned	$+0.1179$	0.004	0.035	-0.0000	0.922	1.000
XGB-64	-0.0129	0.131	0.422	-0.0052	0.105	0.422
XGB-300	-0.0754	0.002	0.035	-0.0249	0.002	0.035
LGBM-64	-0.0567	0.002	0.035	-0.0189	0.002	0.035
LGBM-300	-0.0835	0.002	0.035	-0.0254	0.002	0.035

C.3 Per-Dataset Results for All Models

Tables 8 to 12 give the per-dataset results. Each cell is a mean over ten seeds; the standard deviations of every cell are in the released per-dataset results.

Table 8: PIDT-2S per dataset, mean $\pm$ standard deviation over ten seeds.

Dataset	Accuracy (%)	NLL	Brier score
Adult	84.31 ± 0.78	0.3484 ± 0.0158	0.1100 ± 0.0044
Bank Marketing	89.76 ± 0.42	0.2467 ± 0.0104	0.0739 ± 0.0029
Electricity	78.09 ± 1.00	0.4646 ± 0.0127	0.1512 ± 0.0052
MagicTelescope	86.12 ± 0.44	0.3381 ± 0.0078	0.1033 ± 0.0029
Phishing	93.54 ± 0.55	0.1724 ± 0.0070	0.0494 ± 0.0025
Spambase	93.47 ± 0.57	0.2036 ± 0.0123	0.0534 ± 0.0039
Banknote	94.47 ± 2.19	0.4305 ± 0.0137	0.1256 ± 0.0065
Credit-g	71.60 ± 1.53	0.5733 ± 0.0140	0.1932 ± 0.0063
Phoneme	79.71 ± 0.86	0.4090 ± 0.0080	0.1329 ± 0.0036
QSAR Biodeg	85.12 ± 1.64	0.4656 ± 0.0115	0.1429 ± 0.0053

C.4 Ablations

The tied-logit control and the regulariser sweep use Adult, MagicTelescope and Banknote with seeds $\{0, 1, 2\}$ and the main training protocol.

Table 9: Test accuracy (%) per dataset and model, mean over ten seeds.

Dataset	PIDT-2S	PIDT-1S	PFB	SDT	CART-64	CART-tuned	XGB-64	XGB-300	LGBM-64	LGBM-300
Adult	84.31	84.39	84.39	83.88	84.12	83.95	84.32	85.41	85.49	85.43
Bank Marketing	89.76	89.45	89.45	89.59	89.62	89.63	89.51	90.36	90.16	90.42
Electricity	78.09	77.98	77.98	77.98	80.81	78.22	79.20	84.94	83.97	85.72
MagicTelescope	86.12	86.11	86.11	85.98	84.36	83.62	85.38	87.77	87.00	87.68
Phishing	93.54	93.56	93.56	93.63	94.03	93.49	93.13	96.36	95.57	96.92
Spambase	93.47	93.33	93.33	93.34	91.90	90.88	92.94	95.14	95.05	95.42
Banknote	94.47	93.13	93.13	93.42	98.29	98.44	97.49	99.31	98.87	99.56
Credit-g	71.60	71.35	71.35	71.15	70.45	70.95	73.95	75.85	74.50	74.75
Phoneme	79.71	79.43	79.43	79.10	84.41	82.47	83.05	88.87	88.00	88.79
QSAR Biodeg	85.12	85.64	85.64	85.12	81.71	81.09	84.83	86.68	85.73	86.73

Table 10: Test NLL per dataset and model, mean over ten seeds (lower is better).

Dataset	PIDT-2S	PIDT-1S	PFB	SDT	CART-64	CART-tuned	XGB-64	XGB-300	LGBM-64	LGBM-300
Adult	0.348	0.350	0.350	0.365	0.436	0.401	0.340	0.313	0.320	0.313
Bank Marketing	0.247	0.251	0.251	0.251	0.375	0.353	0.240	0.214	0.218	0.217
Electricity	0.465	0.468	0.468	0.466	0.512	0.548	0.448	0.341	0.375	0.327
MagicTelescope	0.338	0.340	0.340	0.340	0.475	0.490	0.375	0.299	0.322	0.301
Phishing	0.172	0.177	0.177	0.175	0.213	0.219	0.192	0.096	0.131	0.079
Spambase	0.204	0.216	0.216	0.213	0.602	0.414	0.209	0.133	0.151	0.130
Banknote	0.431	0.454	0.454	0.452	0.373	0.388	0.096	0.022	0.055	0.011
Credit-g	0.573	0.577	0.577	0.578	1.922	1.639	0.518	0.499	0.512	0.508
Phoneme	0.409	0.422	0.422	0.422	0.629	0.539	0.371	0.265	0.287	0.265
QSAR Biodeg	0.466	0.480	0.480	0.476	1.861	1.369	0.360	0.317	0.328	0.321

C.5 Training Time

D Implementation Details

Software. PIDT and the differentiable controls are implemented in PyTorch with `torch.cfloat` arithmetic. CART uses scikit-learn (Pedregosa et al., 2011), the boosted references use the `xgboost` and `lightgbm` packages, and data are loaded with the OpenML Python API. The original runs (seeds 0 to 4 on every dataset and seeds 5 to 9 on Adult, Bank Marketing and Electricity) used CPU-only PyTorch 2.4.1. The 350 runs that completed seeds 5 to 9 on the other seven datasets, and all analyses, used Python 3.12 with PyTorch 2.7.1 on the CPU, scikit-learn 1.8.0, XGBoost 3.2.0 and LightGBM 4.6.0; code, hyperparameters, splits and hardware were identical.

Hardware. All runs used one Intel i7-10610U CPU (4 cores, 8 threads) without a GPU. The 1000 training runs of the main sweep take about 4.8 CPU-hours in total and the ablations about 3.9; the analyses run in minutes.

Numerical-stability choices. Gates are clipped to $[\gamma, 1-\gamma]$ with $\gamma = 10^{-2}$ (Eq. 2), which bounds the derivative of the $\sqrt{g_v}$ factors in Eqs. (1) and (3). The branch angles are initialised away from zero so that state mixing is active from the first mini-batch; at zero angles the score is stationary in the angles. These choices are numerical safeguards, not theoretical claims.

Released artefacts. The supplementary code release contains the model and training code, the configurations, metrics and test predictions of every main-sweep run, the per-run metrics of the

Table 11: Test Brier score per dataset and model, mean over ten seeds (lower is better).

Dataset	PIDT-2S	PIDT-1S	PFB	SDT	CART-64	CART-tuned	XGB-64	XGB-300	LGBM-64	LGBM-300
Adult	0.110	0.110	0.110	0.114	0.110	0.111	0.107	0.100	0.101	0.100
Bank Marketing	0.074	0.075	0.075	0.074	0.074	0.077	0.072	0.067	0.067	0.067
Electricity	0.151	0.152	0.152	0.152	0.135	0.150	0.144	0.107	0.116	0.102
MagicTelescope	0.103	0.104	0.104	0.104	0.118	0.122	0.114	0.091	0.097	0.091
Phishing	0.049	0.050	0.050	0.050	0.043	0.046	0.053	0.028	0.035	0.023
Spambase	0.053	0.055	0.055	0.055	0.066	0.072	0.055	0.037	0.039	0.035
Banknote	0.126	0.136	0.136	0.135	0.017	0.016	0.020	0.004	0.010	0.003
Credit-g	0.193	0.195	0.195	0.195	0.212	0.202	0.173	0.165	0.170	0.169
Phoneme	0.133	0.138	0.138	0.138	0.113	0.121	0.119	0.081	0.088	0.081
QSAR Biodeg	0.143	0.149	0.149	0.147	0.140	0.139	0.110	0.096	0.100	0.097

Table 12: Test ECE (15 bins) per dataset and model, mean over ten seeds (lower is better).

Dataset	PIDT-2S	PIDT-1S	PFB	SDT	CART-64	CART-tuned	XGB-64	XGB-300	LGBM-64	LGBM-300
Adult	0.025	0.026	0.026	0.027	0.027	0.020	0.036	0.017	0.022	0.018
Bank Marketing	0.018	0.022	0.022	0.025	0.025	0.021	0.024	0.016	0.017	0.018
Electricity	0.023	0.024	0.024	0.024	0.036	0.030	0.053	0.027	0.038	0.022
MagicTelescope	0.021	0.025	0.025	0.022	0.035	0.027	0.063	0.014	0.035	0.015
Phishing	0.020	0.022	0.022	0.022	0.015	0.015	0.049	0.017	0.036	0.010
Spambase	0.039	0.054	0.054	0.052	0.041	0.027	0.060	0.013	0.033	0.011
Banknote	0.283	0.284	0.284	0.290	0.017	0.016	0.057	0.017	0.031	0.006
Credit-g	0.087	0.090	0.090	0.098	0.142	0.128	0.068	0.052	0.057	0.056
Phoneme	0.043	0.040	0.040	0.040	0.048	0.037	0.051	0.024	0.038	0.020
QSAR Biodeg	0.184	0.198	0.198	0.193	0.107	0.095	0.064	0.056	0.050	0.051

ablations, the PIDT-2S checkpoints used by the mechanism diagnostics, the saved split indices, and the scripts that produce every table. Seeds are fixed to $\{0, \ldots, 9\}$ in the main sweep and $\{0, 1, 2\}$ in the ablations.

Table 13: Tied-logit control: test NLL, mean over three seeds. Tying the component logits makes the rotations invisible in the score (Lemma 2). The submitted version averaged a stale duplicate run into the Banknote PIDT-2S cell (0.4469); this table uses the three runs of the final configuration.

Model	Adult	MagicTelescope	Banknote
PIDT-2S	0.3449	0.3395	0.4316
PIDT-2S, tied logits	0.3555	0.3421	0.4596

Table 14: Regulariser sweep. Each cell is mean test accuracy (%) / mean test NLL over the three datasets and three seeds, with the swept weight at the column value and the other two weights at 10^{-3}.

Swept weight	0	10^{-4}	10^{-3}	10^{-2}
$\lambda_{\mathrm{unitary}}$	88.32 / 0.3717	88.24 / 0.3724	88.20 / 0.3720	88.32 / 0.3727
λ_{usage}	88.32 / 0.3712	88.30 / 0.3704	88.20 / 0.3720	87.88 / 0.3834
λ_{floor}	88.21 / 0.3721	88.21 / 0.3720	88.20 / 0.3720	88.28 / 0.3717

Table 15: Mean training wall clock per run in seconds, averaged over the ten datasets (depth six, CPU). For the differentiable models this covers at most 100 epochs; for the tree references it includes the small validation grids.

Model	Seconds	Model	Seconds
PIDT-2S	83.7	CART-tuned	0.4
PIDT-1S	42.1	XGB-64	0.1
PFB	30.1	XGB-300	0.7
SDT	13.5	LGBM-64	0.1
CART-64	0.1	LGBM-300	0.4

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and Section 1 state three contributions and their scope, each matched to the body: the PIDT-2S architecture (Section 3); the theoretical results, namely Lemmas 1, 2, 3 and 5, Theorems 4 and 7, and Propositions 6 and 8, proved in Appendix A; and the empirical study (Section 6). The abstract states that the experiments use constant branch maps, under which PIDT has the soft-tree function class (Proposition 8), that the NLL gain over the matched controls is small (mean 0.008), and that boosted ensembles remain stronger (Table 3). No state-of-the-art claim is made.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 7 has a Limitations paragraph. The experiments cover only constant branch maps of the form $e^{i\phi}R(\theta)$, so input-dependent maps, the general four-angle family of Eq. (6) and the regime of strict separation from soft trees are not tested; all differentiable models are trained under a fixed 100-epoch budget that binds (Section 6.1); tasks are binary and datasets above $10{,}000$ examples are subsampled; depth, state dimension, γ and angle initialisation are not swept; and there is no comparison with deep tabular or foundation models or with parity baselines. The computational cost is reported in Section 5.2 (PIDT-2S trains 5.4 to 6.5 times slower than SDT), and the poor calibration of all differentiable trees on two small datasets is reported in Section 6.2.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.

- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Each result is stated with its assumptions in Sections 3 and 4 and proved in Appendix A: scalar phase cancellation (Lemma 1), score invisibility under tied logits (Lemma 2), mass conservation (Lemma 3), the containment hierarchy with its depth-one strictness witness for input-dependent branch maps (Theorem 4), the interference decomposition (Lemma 5), the shared-circuit parity construction (Proposition 6), the fixed-architecture Rademacher bound under Assumption 1 (Theorem 7, which uses the contraction step of Bartlett and Mendelson (2002)), and the reduction for constant branch maps (Proposition 8). The scope of each result, including which results need input-dependent branch maps, is stated next to it, and all results are numbered and cross-referenced.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 6.1 specifies the datasets and their OpenML identifiers (Table 6), the subsampling cap, the stratified 80/20 split with a 15% validation split, the train-only preprocessing, every model including the reference CART, XGBoost and LightGBM configurations, the optimiser, regularisation weights, gate clipping, initialisation, early stopping, budget and seeds, and the statistical procedure. Appendix C.1 gives the exact commands for every table, Appendix C.2 the test details and Appendix D the software versions and hardware. The mechanism diagnostics are defined exactly in Section 6.4 and Appendix B.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully

might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: All data are public OpenML datasets (Vanschoren et al., 2014) with fixed identifiers (Table 6). The supplementary code release (MIT licence) contains the model, training and analysis code, the saved split indices, the configurations, metrics and test predictions of every main-sweep run, the per-run metrics of the ablations, and the PIDT-2S checkpoints used by the mechanism diagnostics. Its README explains how to obtain the data through the OpenML API or as files from the OpenML data server and gives the command that produces each table.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 6.1 gives the full protocol: depth $D = 6$, batch size 256, Adam with learning rate 10^{-3} and weight decay 10^{-5}, gradient-norm clipping at 1, early stopping on validation NLL with patience 20, a 100-epoch budget, $\gamma = 10^{-2}$, regularisation weights 10^{-3}, angle initialisation $U(-\pi/4, \pi/4)$ and seeds $\{0, \ldots, 9\}$. The same settings are used for every differentiable model and dataset without per-dataset tuning, and the reference models use the small validation grids listed there. γ is fixed for an analytic reason (Section 5.2) and is not tuned.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Variability comes from ten seeds per dataset, which jointly randomise the split and the initialisation. Paired comparisons use a bootstrap with 2000 resamples of the test set per seed, reported as significant wins and losses at the 95% level (Tables 2 and 3). Across datasets we use two-sided exact Wilcoxon signed-rank tests with the dataset as the unit of analysis, and Appendix C.2 adds Holm-adjusted values over all 18 tests (Table 7). Per-dataset means with standard deviations over seeds are in Table 8. No normality assumption is made.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Appendix D gives the hardware (one Intel i7-10610U CPU, no GPU), the software versions and the compute: the 1000 training runs of the main sweep take about 4.8 CPU-hours, the ablations about 3.9, and the analyses minutes; per-model training times are in Table 15. The project as a whole used more compute than reported, for development runs and earlier versions of the experiments that do not enter the paper.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: The work trains small classifiers on public OpenML tabular data (Table 6), involves no human subjects beyond the original public data collection, and makes no deployment claim.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: Section 7 discusses broader impact. PIDT carries the standard risks of classification systems (dataset bias, calibration error and misread explanations), makes no deployment claim and adds no new misuse channel; on the positive side, exact mass conservation and the explicit interference decomposition (Lemma 5) make its routing easier to inspect.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper releases small decision-tree classifiers trained on public tabular data. No pretrained model, generative model or scraped dataset with a high risk of misuse is released.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All datasets are public OpenML datasets (Vanschoren et al., 2014), identified by ID in Table 6, each with licence information on its OpenML page. Software is credited: scikit-learn (Pedregosa et al., 2011) (BSD licence), XGBoost (Chen and Guestrin, 2016) (Apache 2.0), LightGBM (Ke et al., 2017) (MIT), PyTorch (BSD-style) and the OpenML Python API (BSD). Methods we compare with or build on are cited where used, including SDT (Frosst and Hinton, 2017), the ECE of Guo et al. (2017) and the contraction step of Bartlett and Mendelson (2002).

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The released code is documented by a README covering the directory layout, environment, data acquisition, the command for each table and the format of the per-run artefacts, and it is licensed under MIT. No new dataset is introduced.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper involves no crowdsourcing and no research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No human-subject research is involved, so IRB approval does not apply.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: No large language model is part of the method or of the experimental pipeline (training, evaluation, aggregation or table generation).

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.