GEOHOC: GEOMETRIC HIERARCHICAL OUT-OF-DISTRIBUTION CLASSIFICATION VIA NESTED ANGULAR CAPS

Anonymous authors
Paper under double-blind review

ABSTRACT

Many visual recognition tasks involve both hierarchical class structures and unknown classes in an open world. Hierarchical classification focuses on fine-grained recognition along the class hierarchy, while out-of-distribution recognition concerns when an input no longer belongs to the existing classes. Unifying the two requires jointly modeling the semantic direction and acceptance extent of each class, so that the model can both localize an appropriate semantic branch and determine the proper prediction depth. Based on this perspective, we propose GEOHOC, a geometric framework that jointly models semantic directions and acceptance extents. In a normalized vision–language space, each category node is represented by an angular cap with a learnable center and radius. A joint loss encourages correct branch selection and parent–child containment, while promoting image acceptance by correct caps and rejection by incorrect sibling caps. During inference, center similarity selects a candidate child branch, while a calibrated cap-based acceptance rule determines whether to descend or stop at the current node. We further introduce a known-only calibration scheme to select the stopping offset of this acceptance rule. Experiments on three benchmark datasets demonstrate that GEOHOC achieves the best overall performance among the compared methods, effectively balancing known class recognition and ancestor recovery for unseen classes.

1 INTRODUCTION

Image classification assigns semantic labels to images based on visual content and is a foundational task in computer vision. In standard supervised learning, models learn visual representations and class decision rules from labeled examples and predict from a predefined label set. Advances from convolutional neural networks and vision Transformers to vision–language models have improved visual representation, fine-grained discrimination, and the use of linguistic semantics for recognition (Krizhevsky et al., 2012; He et al., 2016; Dosovitskiy et al., 2021; Radford et al., 2021). Many applications also organize classes from broad concepts to specific ones, motivating hierarchical classification.

Hierarchical classification represents labels as a tree or taxonomy, with leaves denoting specific classes and internal nodes denoting shared superordinate concepts (Silla Jr & Freitas, 2011; Deng et al., 2009). Birds, for example, can be organized by order, family, and species, so each species also belongs to the family and order on its ancestral path. This structure captures semantic inclusion: coarse nodes summarize attributes shared by their descendants, while fine nodes distinguish classes within the same branch. Models can thereby share information across levels, jointly learn coarse branch separation and fine-grained discrimination, and restrict predictions to taxonomy-consistent paths (Bertinetto et al., 2020; Valmadre, 2022). The result is a structured label space with semantics at multiple levels.

In this space, leaves and internal nodes correspond to different semantic granularities and can both serve as predictions. If an image supports the sparrow family but not a particular species among visually similar alternatives, predicting the family still conveys reliable class information. Prior work on multi-level classification, hierarchical selective prediction, and free-grained recognition studies

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

how to adjust prediction depth, balance reliability and specificity, and accommodate supervision and outputs at different granularities (Valmadre, 2022; Goren et al., 2024; Park et al., 2026). Hierarchical prediction therefore also requires choosing the output granularity supported by the available evidence.

In an open world, test images may belong to classes outside the training label set, and open-set recognition and out-of-distribution detection study how to identify such unknown inputs (Scheirer et al., 2012; Bendale & Boult, 2016; Hendrycks & Gimpel, 2016; Liu et al., 2020; Vaze et al., 2022). When the label space is hierarchical, however, an unseen fine-grained class may still belong to a higher-level class already known to the model. For example, an unseen bird species should not be forced into one of the known species, while returning only "unknown" would discard its family-level semantics. A more informative target is therefore the deepest modeled ancestor supported by the evidence, which requires identifying both the appropriate semantic branch and the depth at which prediction should stop (Lee et al., 2018; Lang et al., 2024; Pyakurel & Yu, 2024; 2025; Wallin et al., 2025).

Identifying the appropriate semantic branch and prediction depth requires connecting visual evidence with class semantics at multiple granularities. Multimodal learning offers a natural framework: visual features capture image appearance, while textual class names provide semantic references for fine-grained classes and broader concepts. Vision–language models such as CLIP (Radford et al., 2021) and ALIGN (Jia et al., 2021) align these complementary modalities in a shared representation space. In this shared space, image–text similarity can be used to match visual inputs to class nodes at different levels of the existing taxonomy.

However, identifying the most similar class does not by itself determine whether that class should accept the image. An image from an unseen class may belong to an existing ancestor but none of its known children. This motivates hierarchically organized acceptance regions that capture class semantics and acceptance extents, with parent–child relations expressed through region containment.

Motivated by this perspective, we propose GEOHOC, which connects hierarchical class representation with prediction granularity through geometric acceptance regions. First, we represent each existing class node as an angular cap in the normalized image–text space, with a learnable center and angular radius. The center specifies the class's semantic direction, while the radius defines its acceptance extent. A parent–child containment objective encourages child caps to lie within their parent caps, translating semantic inclusion into geometric containment.

Second, we use this geometry for top-down prediction. Center similarity selects a candidate child branch, and the cap-based acceptance criterion determines whether the evidence supports further specialization; otherwise, prediction returns the current node. The stopping offset is calibrated using only known-class examples. For each calibration example, we evaluate recognition of its annotated class with all candidates retained and recovery of its parent after temporarily masking the annotated class branch. We select the offset to balance these two prediction objectives.

Our main contributions are as follows:

- We propose GEOHOC, a unified geometric framework for hierarchical out-of-distribution classification that jointly represents class semantics and acceptance extents through hierarchical regions.
- We develop geometry-guided inference and known-only calibration to select prediction granularity, balancing known-class recognition and ancestor recovery for unseen inputs.
- On three benchmarks, GEOHOC achieves the best results on key evaluation metrics among the compared methods, with ablation studies supporting the effectiveness of its key components.

2 RELATED WORK

Hierarchical classification and prediction granularity. Hierarchical classification leverages class relationships through coarse-to-fine classifiers or hierarchy-aware training objectives, connecting coarse-grained discrimination with fine-grained recognition (Yan et al., 2015; Bertinetto et al., 2020). Beyond class recognition, related work studies how specific predictions should be, balancing prediction granularity and reliability (Valmadre, 2022; Goren et al., 2024), and explores supervision and prediction at different semantic granularities (Park et al., 2026).

Open-set recognition and hierarchical novelty detection. Open-set recognition studies how to recognize known classes while detecting unknown inputs (Bendale & Boult, 2016; Vaze et al., 2022; Lang et al., 2024). Hierarchical novelty detection further predicts existing ancestor classes for unknown samples rather than simply labeling them as unknown. HND introduces top-down and flat classification schemes and uses a leave-one-out strategy to model unseen classes (Lee et al., 2018). E-HND introduces unknown classes associated with parent nodes and distinguishes known from unknown samples through fine-grained evidence allocation (Pyakurel & Yu, 2024). State-FGOD constructs state-based node representations from the full class hierarchy, exploiting inter-class dependencies to distinguish similar known and unknown classes (Pyakurel & Yu, 2025). ProHOC builds a hierarchical probabilistic model using classification networks trained at different depths, predicting leaf nodes for known samples and the corresponding internal nodes for unknown samples (Wallin et al., 2025).

Vision–language models for hierarchical recognition. CLIP and OpenCLIP provide shared image–text representations for matching visual inputs to class semantics (Radford et al., 2021; Cherti et al., 2023). MaPLe adapts both the visual and textual branches to downstream recognition tasks (Khattak et al., 2023). For hierarchical recognition, CHiLS uses subclass descriptions to make predictions and maps the results to parent classes (Novack et al., 2023), while ProTeCt improves prediction consistency across taxonomic levels through prompt tuning (Wu et al., 2024).

3 Problem Setup

Let $\mathcal{X}$ denote the image space and $\mathcal{Y}$ the set of semantic classes considered in the task, with each class represented by its name. A labeled example is $(x, y) \in \mathcal{X} \times \mathcal{Y}$, where x is an image and y is its class label. The training and test sets are $\mathcal{D}_{\text{tr}} = \{(x_i, y_i)\}_{i=1}^{n} \subseteq \mathcal{X} \times \mathcal{Y}$ and $\mathcal{D}_{\text{test}} = \{(x'_j, y'_j)\}_{j=1}^{m} \subseteq \mathcal{X} \times \mathcal{Y}$, where n and m are the numbers of training and test examples, respectively. Labels may have different semantic granularities across examples.

We define the semantic ancestor indicator $g : \mathcal{Y} \times \mathcal{Y} \to \{0, 1\}$ by

$$g(u, v) = \begin{cases} 1, & u \text{ is a strict subclass of } v, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

The subclass relation is not restricted to direct children: $g(u, v) = 1$ indicates that v is a semantic ancestor of u. This relation can be determined from class knowledge.

For each training label y_i, let $H(y_i) = \{y_i\} \cup \{v \in \mathcal{Y} : g(y_i, v) = 1\}$. Collecting the training classes and their ancestors gives the existing class set $V = \bigcup_{i=1}^{n} H(y_i) \subseteq \mathcal{Y}$. The hierarchy $T = (V, E)$ is available during training and inference and forms a finite rooted forest. Each edge $(p, c) \in E$ is directed from parent p to child c, with

$$E = \big\{(p, c) \in V \times V : g(c, p) = 1,\ \nexists v \in V \text{ such that } g(c, v) = g(v, p) = 1\big\}. \tag{2}$$

Thus, p is an existing ancestor of c, with no other existing class between them.

Let $R_T = \{v \in V : \nexists p \in V \text{ with } (p, v) \in E\}$ be the roots of T. We introduce a virtual node $r \notin \mathcal{Y}$ and define $\widetilde{V} = V \cup \{r\}$, $\widetilde{E} = E \cup (\{r\} \times R_T)$, and $\widetilde{T} = (\widetilde{V}, \widetilde{E})$. The node r is the root of $\widetilde{T}$ and serves as a routing node rather than a semantic class. For $p \in \widetilde{V}$, its children are $\mathrm{Ch}_{\widetilde{T}}(p) = \{c \in \widetilde{V} : (p, c) \in \widetilde{E}\}$. We define depth in the augmented tree by $d_{\widetilde{T}}(r) = 0$ and $d_{\widetilde{T}}(v) = 1 + |\{q \in V : g(v, q) = 1\}|$ for $v \in V$. A training label may be any existing node, not necessarily a leaf.

Test classes may belong to V or to $\mathcal{Y} \setminus V$. For an unseen class $u \in \mathcal{Y} \setminus V$ with an existing ancestor, we select the deepest qualifying node:

$$a_{\widetilde{T}}(u) = \operatorname*{arg\,max}_{v \in V :\, g(u,v)=1} d_{\widetilde{T}}(v). \tag{3}$$

This node is the most specific ancestor of u in the existing forest. If u has no existing ancestor, we set $a_{\widetilde{T}}(u) = r$.

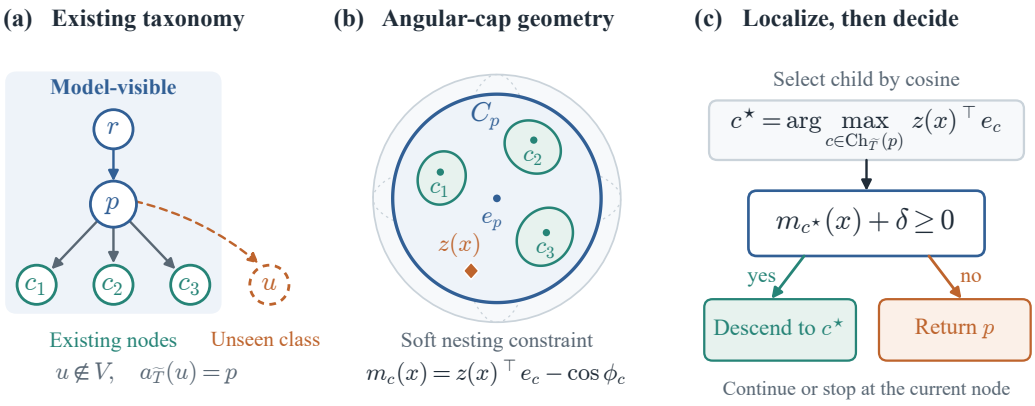


Figure 1: Overview of GEOHOC. (a) Held-out leaves are excluded from the model-visible taxonomy. (b) Each modeled node defines an angular cap; soft nesting regularization encourages child caps to lie within their parent. (c) Cosine similarity selects one child, and its cap margin determines whether to descend or return the current node. The threshold $\theta = -\delta$ is calibrated using normal and masked-child records from known classes, with the model frozen.

Our objective is to learn a node predictor $h : \mathcal{X} \to \widetilde{V}$. For a test example (x, y), the desired prediction satisfies

$$h(x) = \begin{cases} y, & y \in V, \\ a_{\widetilde{T}}(y), & y \notin V. \end{cases} \tag{4}$$

Known-class images should be assigned to their class nodes, while unseen-class images should be assigned to their most specific existing ancestor, or to r if none exists. The ground-truth class y and its ancestor relations determine the evaluation target; the predictor receives only the image x, without access to the unseen class name or its ancestor relations.

4 METHOD

GEOHOC is organized around a single geometric object: a hierarchy of angular acceptance regions. The model first represents every visible node by a center and an angular radius on a shared vision-language sphere. It then learns paths that are discriminative, caps that respect the taxonomy, and boundaries that accept the correct node while excluding nearby siblings. At test time, node centers localize the branch and cap boundaries determine where prediction stops. Figure 1 summarizes the complete procedure.

4.1 HIERARCHICAL ANGULAR CAPS

Node-wise angular regions. Let $f_I : \mathcal{X} \to \mathbb{R}^{512}$ denote the image encoder of OpenCLIP ViT-B/16, including its projection into the shared image–text space. For an image $x \in \mathcal{X}$, we define its normalized representation as

$$z(x) = \frac{f_I(x)}{\|f_I(x)\|_2} \in \mathbb{S}^{511}, \tag{5}$$

where $\mathbb{S}^{511} = \{z \in \mathbb{R}^{512} : \|z\|_2 = 1\}$ is the unit sphere in $\mathbb{R}^{512}$. Each node $v \in V$ has a unit center $e_v \in \mathbb{S}^{511}$ and a learnable angular radius $\phi_v \in (0, \pi)$, defining the angular cap

$$C_v = \{z \in \mathbb{S}^{511} : \mathrm{ang}(z, e_v) \le \phi_v\}, \tag{6}$$

where $\mathrm{ang}(a, b) = \arccos(a^\top b)$ is the angular distance between unit vectors $a, b \in \mathbb{S}^{511}$. The center specifies semantic direction, while the radius specifies acceptance extent. We define the cap margin as

$$m_v(x) = \cos\big(\mathrm{ang}(z(x), e_v)\big) - \cos\phi_v = z(x)^\top e_v - \cos\phi_v, \tag{7}$$

where the second equality follows from $\|z(x)\|_2 = \|e_v\|_2 = 1$. Since cosine is decreasing on $[0, \pi]$, $m_v(x) \ge 0$ if and only if $z(x) \in C_v$. The virtual root r has no center or cap; routing from r uses the centers and caps of its semantic children.

Parent–child geometry. A child class is semantically contained in its parent. We reflect this relation by encouraging $C_c \subseteq C_p$ for each edge $(p, c) \in E$. A sufficient condition is

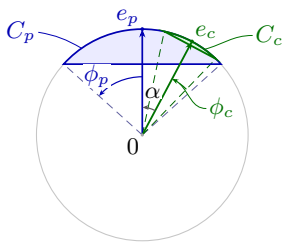
$$\mathrm{ang}(e_p, e_c) + \phi_c + \eta_{\mathrm{nest}} \leq \phi_p, \tag{8}$$

where $\eta_{\mathrm{nest}} \geq 0$ is an angular containment margin. Geometrically, this condition means that the parent's angular radius must accommodate the angular separation between the two centers, the child's angular radius, and the additional margin.

For any $z \in C_c$, the triangle inequality for angular distance gives

$$\begin{aligned} \mathrm{ang}(z, e_p) &\leq \mathrm{ang}(z, e_c) + \mathrm{ang}(e_c, e_p) \\ &\leq \phi_c + \mathrm{ang}(e_c, e_p) \\ &\leq \phi_p - \eta_{\mathrm{nest}} \leq \phi_p. \end{aligned}$$

Thus, whenever equation 8 holds, every point in the child cap also belongs to the parent cap, establishing $C_c \subseteq C_p$.



Learning centers and radii. We freeze the pretrained OpenCLIP weights and insert rank-8 LoRA modules into the attention and MLP projections of both encoders. Let $f_T : \mathcal{Y} \to \mathbb{R}^{512}$ be the adapted text encoder, with normalized text representations $\bar{e}_v = f_T(v)/\|f_T(v)\|_2$ for $v \in V$. Internal centers are initialized from these representations and then optimized as independent unit-norm parameters. For a leaf c with parent p, we use a zero-initialized learnable residual $r_c \in \mathbb{R}^{512}$ to construct

$$e_c = f_{e_p}\left(f_{e_p}^{-1}(\bar{e}_c) + P_{e_p} r_c\right). \tag{9}$$

Here, P_{e_p} projects onto the tangent space at e_p; $f_{e_p}^{-1}$ converts a spherical direction into a tangent vector whose length equals its angular distance from e_p, and f_{e_p} maps the corrected vector back to the sphere. For every node $v \in V$, an unconstrained parameter $\rho_v \in \mathbb{R}$ determines its angular radius:

$$\phi_v = \phi_{\min} + (\pi - \phi_{\min})\sigma(\rho_v), \qquad \phi_{\min} = 0.05, \tag{10}$$

where $\sigma(s) = (1 + e^{-s})^{-1}$, ensuring $\phi_v \in (0.05, \pi)$ in radians. These trainable parameters are optimized using the objective in Section 4.2. Explicit mapping formulas and details appear in Appendix A.

4.2 Learning the Cap Hierarchy

We learn the cap hierarchy through three complementary objectives: selecting the correct child at each parent, encouraging hierarchical containment between parent and child caps, and accepting an image within the correct cap while excluding it from incorrect sibling caps.

For each training example $(x_i, y_i) \in \mathcal{D}_{\mathrm{tr}}$, let $\mathcal{P}_i = \{(p, c) \in \widetilde{E} : c \in H(y_i)\}$ denote the set of edges on the unique path from the virtual root r to y_i. For each $(p, c) \in \mathcal{P}_i$, p is the current parent and c is the correct child for this example. The path ends at the annotated node y_i, which need not be a leaf.

Path discrimination. At each parent p, branch selection uses the similarity between the image representation and the child centers. Let $w_d > 0$ be the path weight at depth d, and define the total path weight for example i as

$$W_i = \sum_{(p,c) \in \mathcal{P}_i} w_{d_{\widetilde{T}}(c)}.$$

Let $s_{iv} = z(x_i)^\top e_v$ denote the cosine similarity between image x_i and the center of node v. The path discrimination loss is

$$\mathcal{L}_{\mathrm{hier}} = -\frac{1}{n}\sum_{i=1}^{n} \frac{1}{W_i} \sum_{(p,c) \in \mathcal{P}_i} w_{d_{\widetilde{T}}(c)} \log \frac{\exp\big((s_{ic} - \gamma)/\tau\big)}{\exp\big((s_{ic} - \gamma)/\tau\big) + \sum_{c' \in \mathrm{Ch}_{\widetilde{T}}(p) \setminus \{c\}} \exp\big(s_{ic'}/\tau\big)}. \tag{11}$$

Here, $\tau > 0$ is the temperature and $\gamma \geq 0$ is the classification margin. The loss first takes a weighted average over each supervised path and then averages across training examples. This objective encourages the model to select the correct child c at each parent p, following the correct semantic path through the taxonomy.

Hierarchical containment. To align geometric containment with the taxonomy, we penalize violations of the parent–child containment condition. We group constrained edges by the depth of their child:

$$\mathcal{E}_d = \big\{(p,c) \in E : d_{\widetilde{T}}(c) = d\big\}, \qquad \mathcal{D}_E = \{d : \mathcal{E}_d \neq \varnothing\}.$$

The hierarchical containment loss is

$$\mathcal{L}_{\text{nest}} = \frac{1}{|\mathcal{D}_E|} \sum_{d \in \mathcal{D}_E} \frac{1}{|\mathcal{E}_d|} \sum_{(p,c) \in \mathcal{E}_d} \big[\text{ang}(e_p, e_c) + \phi_c + \eta_{\text{nest}} - \phi_p\big]_+, \tag{12}$$

where $[q]_+ = \max(q, 0)$ and $\eta_{\text{nest}} \geq 0$ is the angular containment margin defined above. Averaging first within each depth and then across depths ensures that all nonempty depths receive equal weight in the geometric constraint. This loss encourages each child cap to lie within its parent cap, translating semantic parent–child relations into geometric containment.

Cap discrimination. Having the highest center similarity does not ensure that an image lies inside the correct cap or outside its sibling caps. To learn discriminative acceptance boundaries, define the incorrect sibling set for each edge (p, c) as

$$\mathcal{S}_p(c) = \text{Ch}_{\widetilde{T}}(p) \setminus \{c\},$$

and use the smooth penalty

$$\zeta_\beta(q) = \beta \log(1 + \exp(q/\beta)), \qquad \beta > 0,$$

where β controls smoothness. Using the cap margin $m_v(x) = z(x)^\top e_v - \cos\phi_v$ defined above, we write

$$\mathcal{L}_{\text{cap}} = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{|\mathcal{P}_i|} \sum_{(p,c) \in \mathcal{P}_i} \frac{\zeta_\beta\big(\kappa - m_c(x_i)\big) + \mathbb{I}[\mathcal{S}_p(c) \neq \varnothing]\, \zeta_\beta\big(\kappa + \max_{c' \in \mathcal{S}_p(c)} m_{c'}(x_i)\big)}{1 + \mathbb{I}[\mathcal{S}_p(c) \neq \varnothing]}, \tag{13}$$

where $\mathbb{I}[\mathcal{S}_p(c) \neq \varnothing]$ indicates whether incorrect siblings exist. Here, $\kappa = 0.02$ is the cap-discrimination margin. When incorrect siblings exist, we average the correct-cap acceptance term and the incorrect-sibling rejection term; otherwise, only the acceptance term is retained. This loss pushes image representations into the correct cap and away from incorrect sibling caps.

Joint training objective. The overall training objective is

$$\mathcal{L}_{\text{train}} = \lambda_{\text{hier}}\mathcal{L}_{\text{hier}} + \lambda_{\text{nest}}\mathcal{L}_{\text{nest}} + \lambda_{\text{cap}}\mathcal{L}_{\text{cap}}, \tag{14}$$

where the three loss weights are specified in Appendix D.3. The three losses govern branch selection, parent–child containment, and the position of image representations relative to cap boundaries, respectively, jointly learning a structured and discriminative cap hierarchy. No unseen-class images, unseen-class text, or pseudo-open examples are used in training.

4.3 Recursive Prediction with Known-Only Calibration

Localize with centers, stop with caps. For an input image x, inference starts at the virtual root r of the augmented tree $\widetilde{T}$ and proceeds from top to bottom. At the current non-leaf node p, we select the most plausible semantic branch among its children by center similarity:

$$c^\star(p, x) = \arg\max_{c \in \text{Ch}_{\widetilde{T}}(p)} z(x)^\top e_c. \tag{15}$$

The radius does not affect which child is selected. We then use the selected child's cap margin to determine whether to descend. Recall that $m_v(x) = z(x)^\top e_v - \cos\phi_v$ measures the input's position relative to node v's cap boundary: positive values indicate the interior, negative values indicate the exterior, and zero corresponds to the boundary.

We use the stopping offset $\delta \in \mathbb{R}$ as a tunable hyperparameter that adjusts the cap acceptance threshold. Its value is selected using the calibration procedure described below. Let $c^\star = c^\star(p, x)$. The recursive predictor is

$$h_p(x; \delta) = \begin{cases} p, & m_{c^\star}(x) + \delta < 0, \\ c^\star, & m_{c^\star}(x) + \delta \geq 0 \text{ and } c^\star \text{ is a leaf}, \\ h_{c^\star}(x; \delta), & m_{c^\star}(x) + \delta \geq 0 \text{ and } c^\star \text{ is internal}. \end{cases} \tag{16}$$

Table 1: Evaluation protocols and run-specific stopping offsets. Each offset maximizes the known-only calibration objective $J(\delta)$ for the corresponding run. Columns 0–4 index class folds for CIFAR-100 and CUB-200, and seeds for Stanford Cars.

Dataset	Known / novel	Novel target	Repetitions	Stopping offset δ				
				0	1	2	3	4
CIFAR-100	80 / 20	Coarse class	5 folds	0.50	0.50	0.50	0.50	0.50
CUB-200	160 / 40	Family	5 folds	0.32	0.33	0.33	0.34	0.34
Stanford Cars	166 / 30	Body type	5 seeds	0.26	0.24	0.24	0.24	0.24

The final prediction is $h(x;\delta) = h_r(x;\delta)$. Under this recursion, rejecting the selected child returns the current node p without trying another sibling. An accepted child is returned if it is a leaf; otherwise, recursion continues from that child. Rejection at the first step returns r. Center similarity determines which child is selected, while the cap margin and stopping hyperparameter δ determine whether inference enters that child, thereby determining the semantic level of the final prediction. A flowchart and pseudocode are provided in Appendix B.

Known-only stopping-offset selection. After training, we freeze the model and select the stopping offset δ using a known-class calibration set $\mathcal{D}_{\text{cal}} \subseteq \mathcal{X} \times Y$ disjoint from the training set, with annotations at arbitrary class nodes. For the same image, normal prediction retains all candidates and uses the annotated class c as its target; masked-child prediction temporarily removes c from the candidate set of its parent p and uses p as its target. The latter uses only examples whose annotated class has a parent and at least one sibling remaining after masking, testing rejection of incorrect branches rather than forced stopping with no candidates. Let $A_{\text{K}}(\delta)$ denote normal accuracy macro-averaged over annotated classes, and $A_{\text{LOO}}(\delta)$ denote masked-child accuracy macro-averaged over target parents. When eligible masked-child examples exist, we select

$$J(\delta) = \frac{A_{\text{K}}(\delta) + A_{\text{LOO}}(\delta)}{2}, \qquad \delta_0 = \arg\max_{\delta} J(\delta). \tag{17}$$

Full statistical definitions and eligibility conditions are provided in Appendix C.

5 Experiments

5.1 Experimental Setup

Datasets and repetitions. We evaluate GeoHOC on CIFAR-100, CUB-200-2011, and Stanford Cars, with each novel class evaluated against its retained direct parent. Table 1 summarizes the class splits, novel targets, repetitions, and stopping offsets. Detailed hierarchy construction and data partitions are provided in Appendix D.1.

Baselines and fair comparison. We compare against ProHOC (Wallin et al., 2025), State-FGOD (Pyakurel & Yu, 2025), E-HND (Pyakurel & Yu, 2024), HSC transfer (Goren et al., 2024), and HND TD+LOO (Lee et al., 2018). Within each dataset, all methods share the same known/novel class partitions and known optimization/calibration image split. Novel-class images, names, and text prototypes are excluded from training and calibration.

Evaluation metrics. We report Known BAcc, Novel-parent BAcc, MixBAcc, and AUROC. Known BAcc macro-averages recall over known leaf classes and requires prediction of the correct leaf node. Novel-parent BAcc macro-averages recall over the target parent classes of novel samples and requires prediction of the exact target parent; predicting a higher ancestor is not counted as correct. MixBAcc is the equally weighted average of Known BAcc and Novel-parent BAcc. AUROC measures how well a continuous detection score distinguishes known from novel samples. Each metric is computed separately for each run, and we report the mean and sample standard deviation over five runs.

Implementation details. We use OpenCLIP ViT-B/16 pretrained on LAION-2B, with dual-encoder LoRA adaptation and learnable geometric parameters as described in Section 4. Training uses

Table 2: Hierarchical open-set results (mean ± sample standard deviation, %). CIFAR-100 and CUB-200 use five class folds; Stanford Cars uses five seeds. Within each dataset, the highest and second-highest reported means in each metric are **boldfaced** and underlined, respectively.

Method	Known ↑	Novel-parent ↑	Mix ↑	AUROC ↑
CIFAR-100				
ProHOC-R50	75.74 ± 1.05	17.13 ± 1.79	46.43 ± 0.80	80.42 ± 1.10
State-FGOD	80.79 ± 1.22	20.77 ± 5.49	50.78 ± 2.54	**86.60** ± 1.42
HSC transfer	80.70 ± 1.57	7.01 ± 1.13	43.85 ± 0.56	80.55 ± 1.15
E-HND	69.69 ± 8.21	2.13 ± 0.90	35.91 ± 3.98	68.54 ± 3.16
HND TD+LOO	72.35 ± 0.92	5.31 ± 0.93	38.83 ± 0.59	73.78 ± 0.88
GeoHOC	**87.64** ± 1.12	**26.87** ± 6.28	**57.25** ± 2.74	82.77 ± 1.24
CUB-200				
ProHOC-R50	61.58 ± 0.86	**33.07** ± 2.58	47.32 ± 1.20	78.22 ± 1.09
State-FGOD	68.76 ± 0.68	16.57 ± 1.99	42.67 ± 1.23	75.05 ± 1.60
HSC transfer	**71.70** ± 2.24	18.22 ± 1.74	44.96 ± 1.24	**78.53** ± 1.08
E-HND	65.59 ± 0.66	17.97 ± 2.44	41.78 ± 1.30	65.88 ± 1.56
HND TD+LOO	43.47 ± 0.76	0.77 ± 0.31	22.12 ± 0.53	66.81 ± 0.69
GeoHOC	68.91 ± 1.52	28.47 ± 3.95	**48.69** ± 1.25	72.03 ± 1.79
Stanford Cars				
ProHOC-R50	73.98 ± 0.57	**42.39** ± 2.28	58.18 ± 1.21	84.28 ± 0.46
State-FGOD	46.15 ± 0.58	42.03 ± 0.71	44.09 ± 0.30	68.33 ± 0.51
HSC transfer	**84.87** ± 0.95	17.98 ± 2.79	51.42 ± 1.08	**85.06** ± 0.62
E-HND	30.24 ± 5.95	34.04 ± 9.32	32.14 ± 5.97	58.05 ± 1.92
HND TD+LOO	64.08 ± 1.19	0.35 ± 0.32	32.22 ± 0.61	79.33 ± 0.33
GeoHOC	81.70 ± 0.61	36.19 ± 2.56	**58.94** ± 1.00	82.46 ± 1.57

AdamW and alternates between updating the visual LoRA together with the geometric parameters and updating the text LoRA. Full training configurations are provided in Appendix D.3.

5.2 Main Results

As shown in Table 2, GeoHOC achieves the highest mean Mix on all three datasets, outperforming the strongest baseline in Mix by 6.47, 1.37, and 0.76 percentage points on CIFAR-100, CUB-200, and Stanford Cars, respectively. Compared with baselines, GeoHOC improves Mix across all three datasets, indicating that GeoHOC demonstrates a more favorable balance between known-class recognition and novel-parent recovery, highlighting its advantage in final hierarchical node prediction. These results support jointly modeling class recognition and prediction granularity: hierarchical structure not only facilitates fine-grained discrimination among known classes but also helps preserve recognizable ancestor-level semantics when the fine-grained class is unseen.

5.3 Ablation Studies and Analysis

Table 3 shows that removing the path discrimination loss $\mathcal{L}_{\text{hier}}$, hierarchical containment loss $\mathcal{L}_{\text{nest}}$, or cap discrimination loss $\mathcal{L}_{\text{cap}}$ reduces the mean Mix, indicating that all three objectives contribute to overall performance. Likewise, freezing the internal-node centers or angular radii lowers the mean Mix, supporting the effectiveness of learnable centers and radii. These results suggest that the components help balance known-class recognition and novel-parent recovery. We compare against two hierarchical cosine classifiers trained using only $\mathcal{L}_{\text{hier}}$, with either a global stopping threshold or depth-wise thresholds. The full model achieves a higher mean Mix than both baselines under the same calibration objective, supporting the benefit of learned cap geometry over these shared-threshold alternatives. Details of both alternative stopping rules appear in Appendix E.1.

5.4 Geometry and Prediction Trade-offs

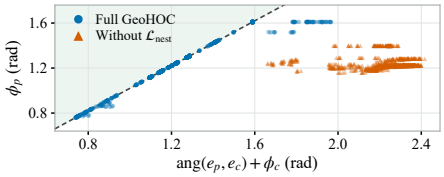
Parent–child geometry. On CUB-200, we examine the raw-cap containment condition using $v_{p,c} = \text{ang}(e_p, e_c) + \phi_c + \eta_{\text{nest}} - \phi_p$. We report the fraction of edges with $v_{p,c} > 10^{-5}$ and the mean positive excess $[v_{p,c}]_+$.

Table 3: CUB-200 ablations (five-fold mean ± sample standard deviation, %). All variants use per-fold known-only calibration with the same selection objective. The best and second-best results according to the primary metric (Mix) are **boldfaced** and underlined, respectively.

Variant	Known ↑	Novel-parent ↑	Mix (primary) ↑
Training objectives			
Without $\mathcal{L}_{\text{hier}}$	0.04 ± 0.03	8.32 ± 0.53	4.18 ± 0.26
Without $\mathcal{L}_{\text{nest}}$	71.79 ± 1.74	22.32 ± 2.93	47.05 ± 0.64
Without $\mathcal{L}_{\text{cap}}$	55.15 ± 1.27	37.38 ± 1.49	46.27 ± 0.71
Node parameters			
Frozen internal centers	40.35 ± 1.10	17.18 ± 3.77	28.77 ± 1.77
Frozen radii	70.22 ± 0.99	21.57 ± 0.92	45.90 ± 0.51
Alternative stopping rules			
Cosine + global threshold	72.03 ± 1.92	1.39 ± 0.39	36.71 ± 0.94
Cosine + depth-wise thresholds	69.38 ± 1.32	25.66 ± 2.68	$\underline{47.52} \pm 0.88$
Full **GeoHOC**	68.91 ± 1.52	28.47 ± 3.95	$\mathbf{48.69} \pm 1.25$

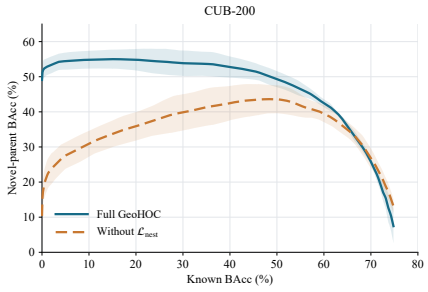
Variant	Violation (%) ↓	Excess (rad) ↓
Without $\mathcal{L}_{\text{nest}}$	100.00 ± 0.00	0.90351 ± 0.00075
Full **GeoHOC**	39.41 ± 2.15	0.04019 ± 0.00050

Table 4: Raw-cap geometry on CUB-200.



Both metrics are averaged within child depths and equally across depths, excluding virtual-root edges. Table 4 reports five-fold means and sample standard deviations. The accompanying plot pools edges across folds; points on or above the dashed line satisfy containment. Nesting reduces both the violation rate and mean excess, improving geometric consistency without enforcing strict containment on every edge.

Recognition–fallback trade-off. With checkpoints fixed, we sweep $\delta \in [-0.5, 0.5]$ in steps of 0.01. Lines and shading show five-fold means and sample deviations at matched Known BAcc. Over a broad Known BAcc range, the full model achieves higher Novel-parent BAcc at matched Known BAcc. This suggests nesting can improve the recognition–fallback trade-off beyond merely shifting the stopping threshold. These curves describe test-set trade-offs; interpolation details are provided in Appendix E.3.



6 Conclusion

We presented GEOHOC, which connects semantic branch localization with prediction-depth selection through learned hierarchical acceptance regions, grounding class representation and parent fallback in a shared geometry. Each node is represented by an angular cap with a learnable center and radius: center similarity localizes a candidate branch, while cap boundaries determine whether the image evidence supports a more specific prediction. Experiments on three datasets demonstrate that GEOHOC effectively balances known-leaf recognition and novel-parent recovery, achieving the highest mean MixBAcc among the compared methods on all three datasets. Ablation studies support the contributions of the training objectives, learnable geometric parameters, and inference method to model performance. Future work will address images containing objects from multiple classes (Zolfi et al., 2024) by extending this geometric framework to open-world object detection (Joseph et al., 2021) and instance segmentation (Wang et al., 2022), jointly localizing or segmenting individual objects and predicting their classes at appropriate semantic granularities.

AI USE STATEMENT

In this work, generative AI tools were used to assist with writing parts of the code, literature searches, and language editing. The authors carefully review AI-assisted content and take full responsibility for the content and accuracy of the entire paper.

REFERENCES

Abhijit Bendale and Terrance E Boult. Towards open set deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1563–1572, 2016.

Luca Bertinetto, Romain Mueller, Konstantinos Tertikas, Sina Samangooei, and Nicholas A Lord. Making better mistakes: Leveraging class hierarchies with deep networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12503–12512. IEEE, 2020.

Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818–2829. IEEE, 2023.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. URL `https://arxiv.org/abs/2010.11929`.

Shani Goren, Ido Galil, and Ran El-Yaniv. Hierarchical selective classification. *Advances in Neural Information Processing Systems*, 37:111047–111073, 2024.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International conference on machine learning*, pp. 4904–4916. PMLR, 2021.

KJ Joseph, Salman Khan, Fahad Shahbaz Khan, and Vineeth N Balasubramanian. Towards open world object detection. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5826–5836. IEEE, 2021.

Muhammad Uzair Khattak, Hanoona Rasheed, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. Maple: Multi-modal prompt learning. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19113–19122. IEEE, 2023.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Nico Lang, Vésteinn Snæbjarnarson, Elijah Cole, Oisin Mac Aodha, Christian Igel, and Serge Belongie. From coarse to fine-grained open-set recognition. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17804–17814. IEEE, 2024.

Kibok Lee, Kimin Lee, Kyle Min, Yuting Zhang, Jinwoo Shin, and Honglak Lee. Hierarchical novelty detection for visual object recognition. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1034–1042. IEEE, 2018.

Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. *Advances in neural information processing systems*, 33:21464–21475, 2020.

Zachary Novack, Julian McAuley, Zachary Chase Lipton, and Saurabh Garg. Chils: Zero-shot image classification with hierarchical label sets. In *International Conference on Machine Learning*, pp. 26342–26362. PMLR, 2023.

Seulki Park, Zilin Wang, and Stella X Yu. Free-grained hierarchical visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 32767–32776, 2026.

Spandan Pyakurel and Qi Yu. Hierarchical novelty detection via fine-grained evidence allocation. In *Forty-first International Conference on Machine Learning*, 2024.

Spandan Pyakurel and Qi Yu. Learning state-based node representations from a class hierarchy for fine-grained open-set detection. In *ICML*, 2025.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Walter J Scheirer, Anderson de Rezende Rocha, Archana Sapkota, and Terrance E Boult. Toward open set recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(7): 1757–1772, 2012.

Carlos N Silla Jr and Alex A Freitas. A survey of hierarchical classification across different application domains. *Data mining and knowledge discovery*, 22(1):31–72, 2011.

Jack Valmadre. Hierarchical classification at multiple operating points. *Advances in neural information processing systems*, 35:18034–18045, 2022.

Sagar Vaze, Kai Han, Andrea Vedaldi, and Andrew Zisserman. Open-set recognition: A good closed-set classifier is all you need. In *International conference on learning representations*, 2022.

Erik Wallin, Fredrik Kahl, and Lars Hammarstrand. Prohoc: Probabilistic hierarchical out-of-distribution classification via multi-depth networks. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20612–20621. IEEE, 2025.

Weiyao Wang, Matt Feiszli, Heng Wang, Jitendra Malik, and Du Tran. Open-world instance segmentation: Exploiting pseudo ground truth from learned pairwise affinity. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4412–4422. IEEE, 2022.

Tz-Ying Wu, Chih-Hui Ho, and Nuno Vasconcelos. Protect: Prompt tuning for taxonomic open set classification. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16531–16540. IEEE, 2024.

Zhicheng Yan, Hao Zhang, Robinson Piramuthu, Vignesh Jagadeesh, Dennis DeCoste, Wei Di, and Yizhou Yu. Hd-cnn: hierarchical deep convolutional neural networks for large scale visual recognition. In *Proceedings of the IEEE international conference on computer vision*, pp. 2740–2748, 2015.

Alon Zolfi, Guy Amit, Amit Baras, Satoru Koda, Ikuya Morikawa, Yuval Elovici, and Asaf Shabtai. Yolood: Utilizing object detection concepts for multi-label out-of-distribution detection. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5788–5797. IEEE, 2024.

A Center and Radius Parameterization

A.1 Vision–language adaptation and internal centers

We freeze the pretrained OpenCLIP image and text encoders and insert trainable rank-8 LoRA modules into their attention and MLP projections. Let $f_T : \mathcal{Y} \to \mathbb{R}^{512}$ denote the adapted text encoder. Each class node $v \in V$ has a normalized text representation

$$\bar{e}_v = \frac{f_T(v)}{\|f_T(v)\|_2} \in \mathbb{S}^{511}. \tag{A.1}$$

The text representations change with the text-side LoRA parameters. For each internal node, we initialize its center as

$$e_v^{(0)} = \bar{e}_v^{(0)}, \tag{A.2}$$

where (0) denotes initialization. Thereafter, the center is optimized as an independent parameter and normalized to unit length, rather than regenerated from the current text representation.

A.2 Sphere–tangent maps and leaf-center refinement

For a reference direction $a \in \mathbb{S}^{511}$, define its tangent space as

$$T_a\mathbb{S}^{511} = \{w \in \mathbb{R}^{512} : a^\top w = 0\}.$$

We use a fixed geometric mapping $f_a : T_a\mathbb{S}^{511} \to \mathbb{S}^{511}$ to convert a tangent vector into a spherical direction:

$$f_a(w) = \cos(\|w\|_2)a + \frac{\sin(\|w\|_2)}{\|w\|_2}w, \qquad w \neq 0, \tag{A.3}$$

with $f_a(0) = a$ by continuous extension. The formula is defined for all tangent vectors. Restricting f_a to $\|w\|_2 < \pi$ gives a bijection onto $\mathbb{S}^{511} \setminus \{-a\}$; we denote the inverse of this restriction by f_a^{-1}. For $b \in \mathbb{S}^{511} \setminus \{-a\}$, let $\theta = \arccos(a^\top b) \in [0, \pi)$. Its tangent representation is

$$f_a^{-1}(b) = \frac{\theta}{\sin\theta}\left(b - (a^\top b)a\right), \qquad 0 < \theta < \pi, \tag{A.4}$$

with $f_a^{-1}(a) = 0$. Since $\|b - (a^\top b)a\|_2 = \sin\theta$, this mapping satisfies

$$\left\|f_a^{-1}(b)\right\|_2 = \theta = \mathrm{ang}(a, b). \tag{A.5}$$

On the unit sphere, θ is the length of the shortest arc from a to b. Thus, the tangent vector preserves the arc length from the reference point. Both f_a and f_a^{-1} are explicit geometric operations, not additional learned networks.

For a leaf c with parent p, we initialize $r_c \in \mathbb{R}^{512}$ to zero and project it onto the tangent space at the parent center:

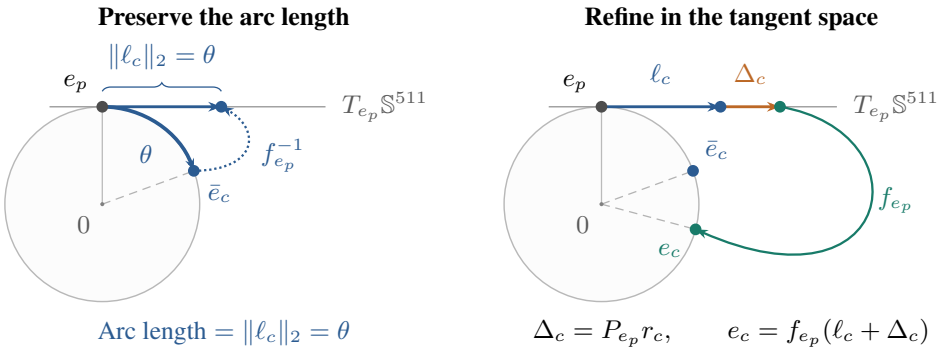
$$\Delta_c = P_{e_p} r_c = r_c - (e_p^\top r_c)e_p, \qquad e_p^\top \Delta_c = 0. \tag{A.6}$$

Writing $\ell_c = f_{e_p}^{-1}(\bar{e}_c)$, the refined leaf center is

$$e_c = f_{e_p}(\ell_c + \Delta_c) = f_{e_p}\left(f_{e_p}^{-1}(\bar{e}_c) + P_{e_p} r_c\right). \tag{A.7}$$

The zero residual gives $e_c = \bar{e}_c$ at initialization, provided $\bar{e}_c \neq -e_p$. During training, the center is recomputed from the current text representation, parent center, and residual. It remains on the unit sphere because $\ell_c + \Delta_c$ is tangent at e_p.

The left sketch highlights the equal lengths of the blue spherical arc and its straight tangent representation. The right sketch adds Δ_c in the same tangent space and maps the sum back to the sphere. Tangent vectors are translated to start at e_p for display; the curved connecting arrows indicate mappings, not projection paths.



A.3 Learnable angular radii

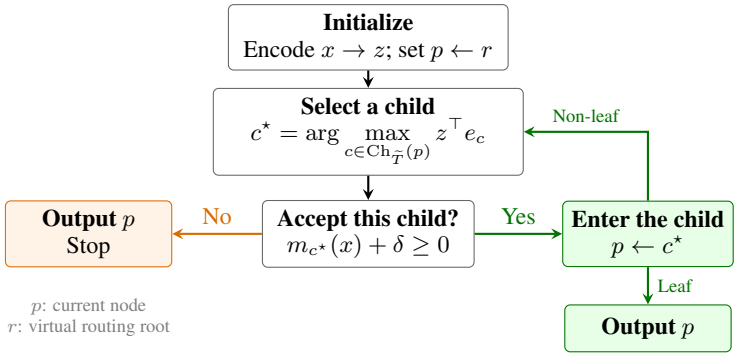
For each node $v \in V$, we learn an unconstrained scalar $\rho_v \in \mathbb{R}$ and use

$$\phi_v = \phi_{\min} + (\pi - \phi_{\min})\sigma(\rho_v), \qquad \phi_{\min} = 0.05, \tag{A.8}$$

where $\sigma(s) = (1 + e^{-s})^{-1}$ is the sigmoid function. Hence $\phi_v \in (0.05, \pi)$ throughout optimization, with angles measured in radians. The trainable parameters comprise both encoders' LoRA parameters, internal centers, leaf residuals, and radius parameters; they are learned using the objectives in the main text.

B Inference Procedure

This section provides a visual summary and pseudocode for the inference procedure described in Section 4.3. Starting from the virtual routing root r, the model first selects the most plausible child by center similarity, and then uses the selected child's cap margin to decide whether to stop at the current node or descend further. Thus, center directions determine the prediction branch, while cap acceptance determines the prediction depth.



$$m_{c^\star}(x) = z(x)^\top e_{c^\star} - \cos\phi_{c^\star}, \qquad \theta = -\delta.$$

Figure 2: Inference first localizes a semantic branch by center similarity, and then decides whether to descend into the selected child using its cap margin. If the selected child is rejected, prediction stops at the current node. If it is accepted and is a leaf, the leaf is returned; otherwise, the procedure continues recursively.

Figure 2 illustrates the same inference rule as equation 16 in a procedural form. The stopping decision is based only on the selected child: if $m_{c^\star}(x) + \delta < 0$, the algorithm stops at the current node p; otherwise it enters $c^\star$. Equivalently, the child is accepted when $z(x)^\top e_{c^\star} \geq \cos\phi_{c^\star} - \delta$, so the offset δ shifts the effective acceptance threshold.

Algorithm 1 Inference by branch localization and cap-based stopping

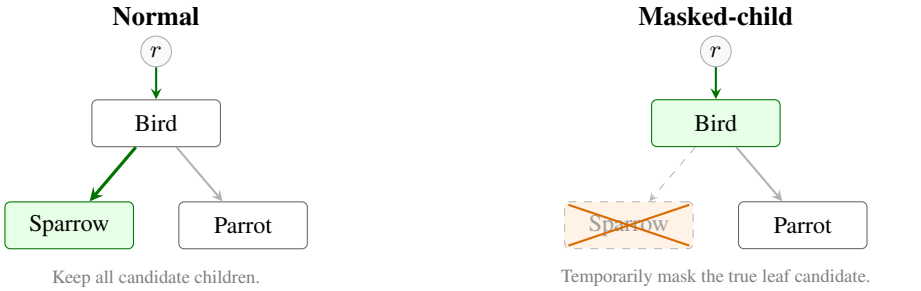
Require: image x, hierarchy $\widetilde{T}$, node centers $\{e_v\}_{v\in V}$, cap radii $\{\phi_v\}_{v\in V}$, stopping offset δ
Ensure: predicted node p

$z \leftarrow z(x)$
$p \leftarrow r$
while True **do**
  $c^\star \leftarrow \arg\max_{c\in \mathrm{Ch}_{\widetilde{T}}(p)} z^\top e_c$
  $m \leftarrow z^\top e_{c^\star} - \cos\phi_{c^\star}$
  if $m + \delta < 0$ **then**
    return p
  else
    $p \leftarrow c^\star$
    if p is a leaf **then**
      return p

C Known-Only Stopping-Offset Calibration

C.1 Example

Figure 3 illustrates the two decoding modes used in known-only calibration. In normal decoding, all candidate children are retained and the target remains the annotated node. In masked-child decoding, the annotated child is temporarily removed from its parent's candidate set, so the target becomes the parent. The first evaluates correct descent to annotated categories, whereas the second evaluates whether the model can stop at the parent when the true child is unavailable.



Target: **Sparrow** contributes to $A_{\mathrm{K}}(\delta)$ Target: **Bird** contributes to $A_{\mathrm{LOO}}(\delta)$

$$J(\delta) = \frac{A_{\mathrm{K}}(\delta) + A_{\mathrm{LOO}}(\delta)}{2}, \qquad \delta_0 = \arg\max_{\delta\in\mathcal{G}} J(\delta), \qquad \mathcal{G} = \{-0.50, -0.49, \ldots, 0.50\}.$$

Figure 3: Illustration of known-only calibration. In normal decoding, all candidate children remain available and the target is the annotated category, contributing to $A_{\mathrm{K}}(\delta)$. In masked-child decoding, the annotated child is temporarily removed from its parent's candidate set, so the target becomes the parent, contributing to $A_{\mathrm{LOO}}(\delta)$. The stopping offset δ_0 is selected by maximizing the average of the two accuracies over the grid $\mathcal{G}$.

C.2 Calibration data and prediction settings

The stopping offset δ controls prediction specificity: increasing it makes child acceptance easier, whereas decreasing it favors earlier stopping. After training, all model parameters are frozen. We use a known-class calibration set

$$\mathcal{D}_{\mathrm{cal}} = \{(x_i, c_i)\}_{i=1}^{M} \subseteq \mathcal{X} \times V, \tag{C.1}$$

where $M \geq 1$ is the number of calibration examples and c_i is the annotated class of image x_i. Calibration images are disjoint from training images. An annotation may be any node in the class forest and need not be a leaf.

Normal prediction. All candidate nodes are retained, and the prediction $h(x_i; \delta)$ is evaluated against the annotated class c_i. All calibration examples contribute to this setting, including those annotated at internal nodes or roots.

Masked-child prediction. For an example whose annotation c_i has a parent p_i, so that $(p_i, c_i) \in E$, we temporarily change the candidate set at p_i to

$$\mathrm{Ch}_{\widetilde{T}}^{(-c_i)}(p_i) = \mathrm{Ch}_{\widetilde{T}}(p_i) \setminus \{c_i\}. \tag{C.2}$$

All other candidate sets remain unchanged. Let $h^{(-c_i)}(x_i; \delta)$ denote the prediction obtained by rerunning the full decoder with this modified candidate set. The correct target is p_i. If c_i is internal, its entire subtree becomes unreachable during this prediction. Masking changes neither the image nor the model parameters and is performed separately for each example, not simultaneously for all annotated classes.

An example contributes to masked-child statistics only if $|\,\mathrm{Ch}_{\widetilde{T}}(p_i)| \geq 2$, so at least one incorrect sibling remains after masking. This tests whether the decoder rejects an incorrect child and stops at the appropriate parent. If the only child were removed, reaching the parent would force termination because no candidate remained, without testing the local cap-based stopping decision. Predictions are evaluated end to end: returning an earlier ancestor or entering an incorrect branch does not count as correct parent recovery. Root-annotated examples have no parent target and are therefore used only for normal prediction. Examples without siblings likewise remain in normal calibration but are excluded from masked-child statistics.

C.3 Balanced accuracy definitions

Define the sample index sets

$$\begin{aligned} \mathcal{I}_c &= \{i \in \{1, \ldots, M\} : c_i = c\}, \\ \mathcal{J}_p &= \{i \in \{1, \ldots, M\} : (p, c_i) \in E,\ |\,\mathrm{Ch}_{\widetilde{T}}(p)| \geq 2\}. \end{aligned} \tag{C.3}$$

The corresponding sets of represented target nodes are

$$\mathcal{C}_{\mathrm{cal}} = \{c \in V : \mathcal{I}_c \neq \varnothing\}, \qquad \mathcal{P}_{\mathrm{cal}} = \{p \in V : \mathcal{J}_p \neq \varnothing\}. \tag{C.4}$$

Here, $\mathcal{I}_c$ groups normal records by annotated class, whereas $\mathcal{J}_p$ groups eligible masked-child records by target parent. Both are formed from the same calibration set; masked-child records use the eligible subset of its examples.

The two balanced accuracies are

$$\begin{aligned} A_{\mathrm{K}}(\delta) &= \frac{1}{|\mathcal{C}_{\mathrm{cal}}|} \sum_{c \in \mathcal{C}_{\mathrm{cal}}} \frac{1}{|\mathcal{I}_c|} \sum_{i \in \mathcal{I}_c} \mathbb{I}[h(x_i; \delta) = c]\,, \\ A_{\mathrm{LOO}}(\delta) &= \frac{1}{|\mathcal{P}_{\mathrm{cal}}|} \sum_{p \in \mathcal{P}_{\mathrm{cal}}} \frac{1}{|\mathcal{J}_p|} \sum_{i \in \mathcal{J}_p} \mathbb{I}\Big[h^{(-c_i)}(x_i; \delta) = p\Big]\,, \end{aligned} \tag{C.5}$$

where $\mathbb{I}[\cdot]$ is the indicator function. Each metric first computes accuracy within a target node and then averages equally across target nodes. Normal records are balanced over annotated classes; masked-child records are pooled by their target parents and balanced over those parents, rather than over the original child classes.

C.4 Grid search and test-time use

Provided that $\mathcal{P}_{\mathrm{cal}} \neq \varnothing$, we use the equally weighted objective

$$J(\delta) = \frac{A_{\mathrm{K}}(\delta) + A_{\mathrm{LOO}}(\delta)}{2}. \tag{C.6}$$

The offset is selected over the grid

$$\mathcal{G} = \{-0.50 + 0.01k : k = 0, 1, \ldots, 100\}, \qquad \delta_0 = \operatorname*{arg\,max}_{\delta \in \mathcal{G}} J(\delta). \tag{C.7}$$

The grid spans $[-0.5, 0.5]$ with step size 0.01. If several offsets attain the maximum, we choose the smallest one. For every candidate offset, both prediction settings use the same frozen model and calibration images. The objective balances correct recognition when the annotated class is available with correct parent recovery when its branch is absent.

D SUPPLEMENTARY EXPERIMENTS

D.1 EXPERIMENTAL SETUP

CIFAR-100. CIFAR-100 contains 100 fine classes grouped into 20 coarse classes. We organize these coarse classes under six fixed semantic roots, yielding a three-level $6 \to 20 \to 100$ hierarchy. In fold $k \in \{0, \ldots, 4\}$, the k-th child of every coarse parent is held out, giving 80 known and 20 novel classes. Each fine class is held out exactly once across the five folds. Each known class contributes 400 official training images for optimization and 100 disjoint images for calibration, resulting in 32,000 optimization and 8,000 calibration images per fold. Evaluation uses all 10,000 official test images, and the target for a novel class is its coarse parent.

CUB-200-2011. CUB-200-2011 contains 200 bird species. We use an Order–Family–Species hierarchy with 13 orders and 38 families, derived from the taxonomy released with State-FGOD (Pyakurel & Yu, 2025). Five class splits each retain 160 known species and hold out 40 novel species, with a known sibling retained for every novel species. Known training images are split within each class into disjoint optimization and calibration partitions, with 960 calibration images per fold. Evaluation uses the complete official test set, and the target for a novel species is its retained family node.

Stanford Cars. Stanford Cars contains 196 vehicle-model classes. We adopt the fixed 166-known/30-novel class split released with State-FGOD (Pyakurel & Yu, 2025). Below a separate virtual routing root, Automobile, body type, and model form three modeled levels. The model-visible hierarchy contains one Automobile node, nine body-type nodes, and 166 known model nodes. Like the other internal nodes, Automobile has a learnable center and angular radius; the virtual root only initiates traversal and has no cap. Known-class images provide 5,486 optimization and 1,379 calibration examples, and evaluation uses all 8,041 official test images. The target for a novel model remains its immediate body-type parent; predicting the higher-level Automobile node does not count as correct fallback. We evaluate five independent training seeds under the fixed class split.

Results are reported as means and sample standard deviations over five class folds for CIFAR-100 and CUB-200, and five training seeds for Stanford Cars.

D.2 EVALUATION METRICS

We evaluate known-leaf recognition, novel-parent recovery, and known-versus-novel discrimination using four metrics. Let $\hat{y}(x)$ denote the predicted node and $\mathbf{1}[\cdot]$ the indicator function.

Known. Known measures balanced accuracy over known leaf classes. Let $\mathcal{K}$ be the set of known classes and $\mathcal{D}_{\mathrm{K}}^{c}$ the known test samples belonging to class c. We compute

$$A_{\mathrm{K}} = \frac{1}{|\mathcal{K}|} \sum_{c \in \mathcal{K}} \frac{1}{|\mathcal{D}_{\mathrm{K}}^{c}|} \sum_{x \in \mathcal{D}_{\mathrm{K}}^{c}} \mathbf{1}[\hat{y}(x) = c].$$

Each known class receives equal weight. A prediction is correct only when it matches the ground-truth leaf; stopping at an internal node is counted as incorrect.

Novel-parent. Novel-parent measures balanced accuracy over the target parents of novel test samples. Let $\mathcal{P}_{\mathrm{N}}$ be the target parents represented in the novel test set, and $\mathcal{D}_{\mathrm{N}}^{p}$ the novel samples whose ground-truth target parent is p. We compute

$$A_{\mathrm{P}} = \frac{1}{|\mathcal{P}_{\mathrm{N}}|} \sum_{p \in \mathcal{P}_{\mathrm{N}}} \frac{1}{|\mathcal{D}_{\mathrm{N}}^{p}|} \sum_{x \in \mathcal{D}_{\mathrm{N}}^{p}} \mathbf{1}[\hat{y}(x) = p].$$

Each target parent receives equal weight, regardless of its number of novel classes or test images. Correctness requires an exact match to the designated parent; predictions at higher ancestors, descendants, or other branches are counted as incorrect.

Mix. Mix assigns equal importance to known-leaf recognition and novel-parent recovery:

$$\mathrm{Mix} = \frac{A_{\mathrm{K}} + A_{\mathrm{P}}}{2}.$$

It is not the accuracy pooled over all test samples.

AUROC. To combine cap geometry with pretrained semantic information, we construct two scores from the standardized angular excess beyond cap boundaries and the negative frozen-CLIP image–text similarity. Using known-class calibration data, we transform them into empirical percentile risks, denoted by $R_{\mathrm{cap}}(x)$ and $R_{\text{frozen-CLIP}}(x)$. The two risks are combined on this common scale as $s(x) = 0.5R_{\mathrm{cap}}(x) + 0.5R_{\text{frozen-CLIP}}(x)$, with the same weights across all datasets. Higher scores indicate stronger evidence of novelty. The area under the receiver operating characteristic curve (AUROC) measures discrimination between known and novel samples, treating novel samples as positives and known samples as negatives. The ROC curve plots the true positive rate against the false positive rate as the detection threshold varies. AUROC is computed over all known and novel test samples using continuous detection scores, with higher values indicating better discrimination.

Aggregation and reporting. All four metrics are reported as percentages, with higher values indicating better performance. We report the mean and sample standard deviation across five class folds for CIFAR-100 and CUB-200, and five random seeds for Stanford Cars. For per-run values $M_1, \ldots, M_5$, the reported statistics are

$$\bar{M} = \frac{1}{5}\sum_{i=1}^{5} M_i, \qquad s_M = \sqrt{\frac{1}{4}\sum_{i=1}^{5}(M_i - \bar{M})^2}.$$

Mix is computed within each run before aggregation; its standard deviation is calculated from the five per-run Mix values.

D.3 Implementation and Optimization

Backbone and preprocessing. Our implementation is based on PyTorch, with mixed-precision training on a single NVIDIA H100 PCIe GPU with 80 GB of memory. We use OpenCLIP ViT-B/16 initialized from the LAION-2B checkpoint `laion2b_s34b_b88k`. Images are processed at 224×224 resolution using the corresponding training and evaluation preprocessing provided by OpenCLIP.

Learnable parameters and initialization. We freeze the original pretrained backbone parameters and introduce rank-8 LoRA with $\alpha_{\mathrm{LoRA}} = 16$ into both the visual and text encoders. Internal-node centers are initialized from normalized class-text features and learned together with angular radii and parent-tangent leaf residuals. All three datasets use three modeled levels, with initial angular radii $(1.20, 0.90, 0.66)$ radians from coarse to fine. Radii are parameterized in $(0.05, \pi)$ radians, and leaf tangent residuals are initialized to zero.

Training objective. Let $\mathcal{L}_{\mathrm{hier}}$ denote the path-weight-normalized classification loss defined in Section 4.2, before multiplication by its loss coefficient. The reported experiments use

$$\mathcal{L}_{\mathrm{train}} = \lambda_{\mathrm{hier}}\mathcal{L}_{\mathrm{hier}} + 0.05\mathcal{L}_{\mathrm{nest}} + 0.05\mathcal{L}_{\mathrm{cap}},$$

with

$$\lambda_{\mathrm{hier}} = \begin{cases} 1.0, & \text{CIFAR-100,} \\ 0.07, & \text{CUB-200 and Stanford Cars.} \end{cases}$$

Path-level weights are $(1, 1, 2)$ across all three datasets, placing greater relative weight on leaf-level classification. The softmax temperature and softplus smoothing scale are both 0.07. The sibling-classification margin is 0.05, while the nesting and cap-discrimination margins are both 0.02. Nesting losses are averaged within each child depth and then equally across nonempty depths; the virtual root is excluded.

Parameter updates and optimizer. Training alternates between two parameter updates. Using cached text features, the first update optimizes visual LoRA and geometric parameters. The second updates text LoRA with detached visual features, after which the text-feature cache is refreshed. Both updates use AdamW with weight decay 10^{-4} and cosine learning-rate annealing. Initial learning rates are 10^{-4} for visual and text LoRA, 5×10^{-5} for internal-node centers and leaf tangent residuals, and 5×10^{-4} for angular radii.

Training schedule. We train CIFAR-100 for 20 epochs with batch size 256, and CUB-200 and Stanford Cars for 90 epochs with batch size 128. Each run is initialized independently from the same pretrained backbone checkpoint and trained under its corresponding class split or random seed.

D.4 Computational cost

We profile CUB-200 image-to-score computation on an NVIDIA H100 PCIe using FP32, 224×224 inputs, and a batch size of 32 for all methods. We report four metrics. Params counts all resident model parameters, and Train. counts the union of parameters updated across training stages, both in millions. GFLOPs reports counted floating-point operations per image, primarily covering convolutions and matrix multiplications, with two operations per multiply-add. Time reports batch latency in milliseconds, using the median of the three round-median batch latencies. Each round uses ten warm-up batches followed by fifty timed batches, with method order shuffled across rounds. The scope includes image encoders and classification heads, but excludes data loading, one-time text embedding computation, and final hierarchical decoding.

As shown in Table 5, GeoHOC contains 151.70M parameters, but approximately 98.63% remain frozen, leaving only 2.07M parameters updated during training. Within the measured image-to-score scope, GeoHOC requires 36.66 GFLOPs per image, comparable to State-FGOD's 36.98 GFLOPs. At a batch size of 32, their measured latencies are 107.48 ms and 99.39 ms, respectively, a difference of approximately 8.1%. These results show that GeoHOC adapts through updates to a small fraction of its parameters while maintaining forward computational costs comparable to State-FGOD.

Table 5: Image-to-score resource costs under the common FP32 protocol on an NVIDIA H100.

Method	Params (M)	Train. (M)	GFLOPs	Time (ms)
GeoHOC	151.70	2.07	36.66	107.48
State-FGOD	86.01	0.21	36.98	99.39
ProHOC-R50	70.96	70.96	24.52	121.32
HSC transfer	23.84	23.84	8.17	38.75
E-HND	42.93	0.43	15.60	74.65
HND TD+LOO	11.33	11.33	3.63	16.34

E Matched-Backbone Comparisons and Geometric Analysis

E.1 Classification-Only Baselines

To distinguish the contribution of geometric modeling from that of the pretrained representation and adaptation, we construct two hierarchical cosine classifiers with global and depth-wise stopping thresholds. Both use the same OpenCLIP initialization, visual and text LoRA, node-center parameterization, data splits, and optimization settings as the full model. Both baselines are trained using only the path classification objective $\mathcal{L}_{\text{hier}}$, with the nesting and cap discrimination losses disabled. Cap radii remain at initialization and participate in neither the training objective nor the stopping rule. For each fold, we train one classification-only model from the pretrained initialization; the two baselines share its weights and differ only in their stopping thresholds.

At the current node p, both baselines select the child with the highest cosine similarity:

$$c^\star = \arg \max_{c \in \mathrm{Ch}_{\bar{T}}(p)} z(x)^\top e_c.$$

The global-threshold baseline accepts this child if $z(x)^\top e_{c^\star} \geq t$. The depth-wise baseline instead requires $z(x)^\top e_{c^\star} \geq t_{\ell(c^\star)}$, where $\ell(c^\star)$ denotes the child's semantic depth. It uses one threshold

Table 6: Depth-wise diagnostics of the raw-cap containment condition on CUB-200. Results are five-fold means ± sample standard deviations. Mean excess is averaged over all edges at the corresponding depth.

Parent–child level	Model	Violation rate (%)	Mean excess (rad)
Order–Family	Without $\mathcal{L}_{\text{nest}}$	100.00 ± 0.00	0.75087 ± 0.00084
Order–Family	Full GEOHOC	28.95 ± 0.00	0.07940 ± 0.00070
Family–Species	Without $\mathcal{L}_{\text{nest}}$	100.00 ± 0.00	1.05615 ± 0.00090
Family–Species	Full GEOHOC	49.88 ± 4.29	0.00098 ± 0.00029

each for Order, Family, and Species, shared by all nodes at the corresponding depth. These are depth-specific rather than node-specific thresholds. If the acceptance condition fails, inference returns the current node without trying another child.

Threshold calibration. Thresholds are not optimized by gradient descent during model training. After training, all network parameters are fixed, and thresholds are selected independently for each fold on known-class calibration data by maximizing

$$J = \frac{A_K + A_{\text{LOO}}}{2}.$$

Here, A_K is class-macro accuracy for known-leaf recognition under normal inference. To compute A_{LOO}, the ground-truth leaf is masked from the routing choices, and its direct parent becomes the prediction target; accuracy is macro-averaged over target parents. This evaluation includes all eligible calibration images whose parent retains another visible child after masking, without filtering for a correctly predicted path prefix.

For the global baseline, we search $t \in \{-1.00, -0.99, \ldots, 1.00\}$ and choose the larger threshold when objective values tie. For the depth-wise baseline, all three thresholds are initialized to the selected global threshold. We then perform three coordinate-search sweeps in Order–Family–Species order. Each update fixes the other two thresholds and searches the same grid to maximize J computed from the complete inference procedure, using the same tie-breaking rule. This is a coordinate search, not an exhaustive search or a guarantee of a globally optimal threshold combination. The depth-wise baseline therefore provides a more flexible stopping rule without introducing additional trained classifier parameters or geometric supervision.

E.2 Depth-Wise Geometric Diagnostics

The main text reports aggregate geometric statistics. Here, we further separate them by parent–child semantic level. For each full-model and no-nesting checkpoint, we read the learned node centers and raw angular radii and compute

$$v_{p,c} = \text{ang}(e_p, e_c) + \phi_c + \eta_{\text{nest}} - \phi_p, \qquad \eta_{\text{nest}} = 0.02.$$

For the edge set $\mathcal{E}_d$ whose children have depth d, we define the condition violation rate and mean positive excess as

$$R_d = \frac{1}{|\mathcal{E}_d|} \sum_{(p,c)\in\mathcal{E}_d} \mathbf{1}[v_{p,c} > 10^{-5}], \qquad V_d = \frac{1}{|\mathcal{E}_d|} \sum_{(p,c)\in\mathcal{E}_d} \max(v_{p,c}, 0).$$

Importantly, V_d averages over all edges, with nonviolating edges contributing zero, rather than averaging only over violating edges. Each fold contains 38 Order–Family edges and 160 Family–known-Species edges. Edges incident to the virtual root are excluded because the root has no cap. The aggregate statistics in the main text average the two depths equally, preventing the more numerous fine-grained edges from dominating the measurements.

The nesting objective reduces both the violation rate and mean excess at both semantic levels. Although some Family–Species edges in the full model still fail the margin-based condition, their average positive excess is small. Violation frequency and magnitude thus provide complementary information and should be interpreted together.

These diagnostics assess a sufficient containment condition with a positive margin on the raw caps. Violating this condition does not, by itself, prove that the child cap is not contained in its parent. The results neither establish strict nesting on every edge nor guarantee containment of the effective acceptance regions after applying the stopping offset.

E.3 Recognition–Fallback Trade-off

Purpose. The stopping offset affects both known-leaf recognition and parent recovery for novel inputs. An increase in Novel-parent accuracy at a single operating point may therefore accompany a decrease in Known accuracy, making it difficult to distinguish an improved trade-off from a change in stopping preference. We compare the full model and the model without nesting at matched Known BAcc to examine how much parent-recovery accuracy each retains at a comparable level of known-class recognition.

Curve construction. Keeping each trained checkpoint fixed, we sweep $\delta \in [-0.5, 0.5]$ in increments of 0.01. Each offset produces a pair of Known BAcc and Novel-parent BAcc values, plotted on the horizontal and vertical axes, respectively. The two models may use different offsets at the same Known level: the comparison matches recognition performance rather than the numerical value of the stopping parameter.

To aggregate the five runs, we first order the scan points by Known BAcc within each run. When multiple points have the same Known value, we retain the highest Novel-parent value and linearly interpolate between adjacent remaining points. We then compute the mean and sample standard deviation of Novel-parent BAcc at common Known values, restricted to the range covered by all five runs of both models, without extrapolation. Lines represent mean performance, while shading indicates variation across class splits rather than confidence intervals. Interpolation only aligns the horizontal coordinates; it neither changes model predictions nor introduces additional evaluated operating points.

Results and interpretation. At low-to-moderate Known levels, the full model's curve lies above that of the model without nesting. Thus, at comparable known-class recognition levels, the full model correctly recovers the target parent for more novel inputs. In this region, the improvement is reflected in the recognition–fallback trade-off, rather than merely in the selection of a more conservative stopping point.

Together with the diagnostics in Appendix E.2, these results provide complementary evidence: the nesting objective improves the geometric relationship between learned parent and child caps and improves parent recovery over part of the recognition range.

E.4 Hyperparameter Sensitivity and Calibration Analysis

Sensitivity to the stopping offset. We keep the full-model checkpoints fixed and sweep $\delta \in [-0.5, 0.5]$ in increments of 0.01. Figure 4 reports Known BAcc, Novel-parent BAcc, and Mix as functions of δ. Lines and shaded bands represent means and sample standard deviations over five class folds for CIFAR-100 and CUB-200 and five random seeds for Stanford Cars.

The selected child is accepted when

$$z(x)^\top e_{c^\star} \geq \cos \phi_{c^\star} - \delta.$$

Increasing δ therefore lowers the threshold for entering the selected child. With the model weights and branch-selection rule unchanged, inputs that previously stopped at internal nodes may proceed to deeper predictions, improving known-leaf recognition until performance saturates.

For novel inputs, a correct prediction requires stopping at the target parent. An overly restrictive acceptance condition may terminate inference before this parent is reached, so moderately increasing δ can reduce premature stopping. Further relaxation, however, may allow inference to proceed beyond the target parent and assign the input to a known leaf. Novel-parent BAcc can consequently vary non-monotonically with δ, while Mix summarizes the resulting balance between known-leaf recognition and parent recovery.

Known-only calibration objective. On the same offset grid, we evaluate known-class accuracy under normal inference, $A_K(\delta)$, parent-recovery accuracy with the ground-truth leaf masked, $A_{\mathrm{LOO}}(\delta)$, and their equally weighted objective:

$$J(\delta) = \frac{A_K(\delta) + A_{\mathrm{LOO}}(\delta)}{2}.$$

The statistical definitions follow Appendix C. Figure 5 reports means and sample standard deviations over the five runs. These quantities are computed on known-class calibration data, separately from the test-set sensitivity curves above.

The two calibration terms capture complementary prediction requirements. A_K rewards reaching the correct known leaf, whereas A_{LOO} rewards returning its parent when that leaf is unavailable. Improving leaf recognition alone therefore need not improve parent recovery; J incorporates both requirements when evaluating the stopping offset. The calibration curves illustrate this joint objective. Their five-run averages are aggregate summaries.

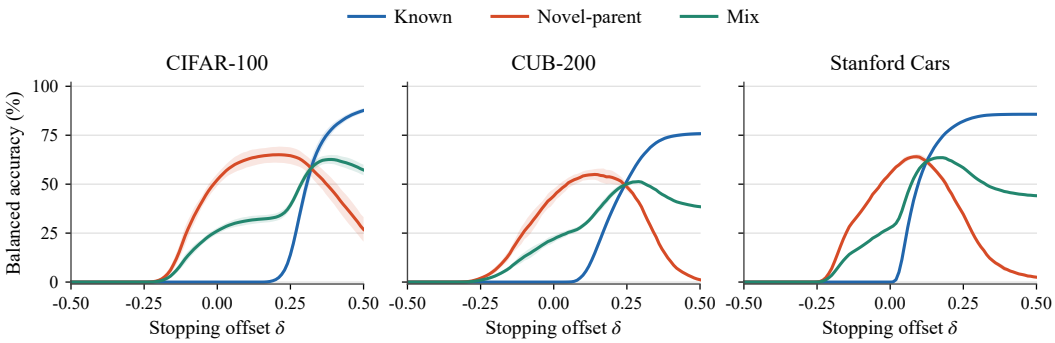


Figure 4: Sensitivity to the stopping offset on fixed full-model checkpoints. Lines and shaded bands show means and sample standard deviations over five class folds for CIFAR-100 and CUB-200 and five seeds for Stanford Cars.

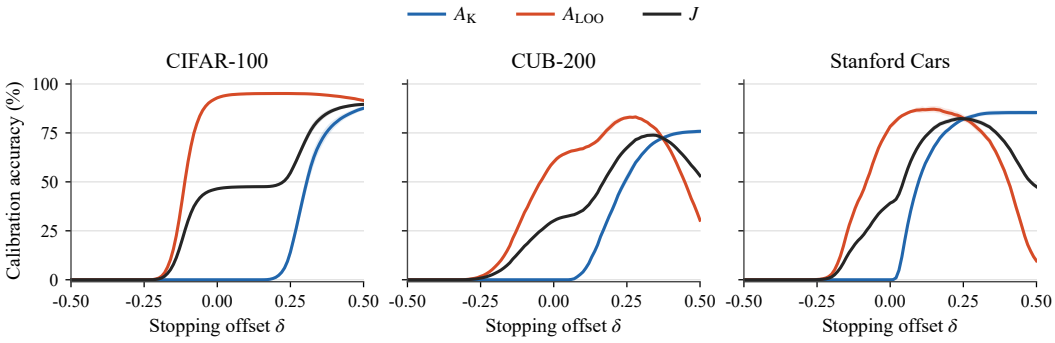


Figure 5: Known-only calibration as a function of the stopping offset. The curves report A_K, A_{LOO}, and $J = (A_K + A_{\mathrm{LOO}})/2$. Lines and shaded bands show means and sample standard deviations over five class folds for CIFAR-100 and CUB-200 and five seeds for Stanford Cars.