Semi-Supervised Hierarchical Open-Set Classification

Supplementary Material

7. Training algorithms

To detail the training algorithms for SemiHOC and the baselines, we first describe the depth-specific networks used for ProHOC predictions and the corresponding target distributions for training these.

The depth-specific network at depth d, parameterized by θ_d, is responsible for classifying the set of categories at that depth, along with any ID leaf classes that occur at shallower depths. Formally, the classification space at depth d is defined as

$$\begin{aligned}\mathcal{C}_d = &\{c \in \mathcal{C} \mid \text{Depth}(c) = d\} \cup \\ &\{c \in \mathcal{C}_{\text{id}} \mid \text{Depth}(c) < d\}\end{aligned} \tag{3}$$

for $d \in \{1, \dots, D\}$ with D being the maximum depth of the hierarchy, $\mathcal{C}$ is the set of all hierarchy nodes, and $\mathcal{C}_{\text{id}}$ denotes the set of ID leaf classes.

To define the target distributions used for training the depth-specific networks, we introduce a mapping function $S_d(c)$ that returns the ancestors or descendants of c present at depth d:

$$S_d(c) = (\text{Anc}^*(c) \cup \text{Desc}(c)) \cap \mathcal{C}_d \tag{4}$$

where $\text{Anc}^*(c)$ denotes the set of ancestors of c (including c itself), and $\text{Desc}(c)$ the set of its descendants of. The target distribution $q_c^d(c')$ over $c' \in \mathcal{C}_d$ with $c \in \mathcal{C}$ is then defined as

$$q_c^d(c') = \begin{cases} \dfrac{1}{|S_d(c)|}, & \text{if } c' \in S_d(c) \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

where $|\cdot|$ denotes the set cardinality.

When c is an ID class (a leaf), q_c^d reduces to a one-hot distribution over $\mathcal{C}_d$, with the unity element at the ancestor of c at depth d. When c is an internal node, q_c^d is one-hot on the ancestors of c for depths $d \leq \text{Depth}(c)$. For deeper levels ($d > \text{Depth}(c)$), the probabilities in q_c^d are spread uniformly over the descendants of c in $\mathcal{C}_d$. The uniform assignment aligns with ProHOC, where the node-local OOD probability is measured through the entropy over the children categories.

SemiHOC is described in Algorithm 2 and the baselines used for comparisons are described in Algorithms 3 to 5. In these algorithms, $\texttt{CE}(\cdot, \cdot)$ denotes the cross-entropy loss. Note that Algorithm 3 is equal to the supervised training in [35]. Any data augmentations applied during student predictions are absorbed into the student parameters θ_d^s; in our case, this corresponds to applying dropout to the input and intermediate features.

Algorithm 2: SemiHOC - training step

Input: Labeled data: $\{(x_i^l, y_i) \mid i \in \{1, \dots, N\}\}$
Unlabeled data with identifiers: $\{(x_i^u, g_i) \mid i \in \{1, \dots, M\}\}$
Depth-specific networks:
students θ_d^s, teachers θ_d^t for $d \in \{1, \dots, D\}$
Current epoch: e
Age cutoffs: T_c for $c \in \mathcal{C}$
Subtree pseudo-label log: SPLlog

```
// Compute the labeled loss
for d ← 1 to D do
    ℓ_d^l ← 0
    ℓ_d^u ← 0
    for i ← 1 to N do
        ℓ_d^l += CE(q^d_{y_i}, p(y|x_i^l; θ_d^s))

// Subtree pseudo-labels for unlabeled data
for i ← 1 to M do
    // Teacher predictions
    p(y|x_i^u) ← ProHOC({p(y|x_i^u; θ_d^t) | d ∈ {1, ..., D}})
    ŷ_i ← ComputeSubtreePLs(p(y|x_i^u)) (following (2))

// Assign pseudo-labels to the log
for i ← 1 to M do
    foreach c where ŷ_{i,c} = 1 do
        if (c, g_i) not in SPLlog then
            SPLlog(c, g_i) ← e
    foreach c where ŷ_{i,c} = 0 do
        if (c, g_i) in SPLlog then
            Delete SPLlog(c, g_i)

// Remove any post-cutoff pseudo-labels
for i ← 1 to M do
    foreach c where ŷ_{i,c} = 1 do
        if SPLlog(c, g_i) > T_c then
            ŷ_{i,c} ← 0

// Add pseudo-labeled data to the unlabeled loss
for i ← 1 to M do
    foreach c where ŷ_{i,c} = 1 do
        foreach d where c ∈ C_d do
            ℓ_d^u += CE(q_c^d, p(y|x_i^u; θ_d^s))
```

Output: The loss for each depth:
$\ell_d = \ell_d^l / N + \ell_d^u / M$ for $d \in \{1, \dots, D\}$
Updated SPLlog

Algorithm 3: Supervised only - training step

Input: Labeled data: $\{(x_i^l, y_i) \mid i \in \{1, \dots, N\}\}$
Depth-specific networks: θ_d for $d \in \{1, \dots, D\}$

```
for d ← 1 to D do
    ℓ_d ← 0
    for i ← 1 to N do
        ℓ_d += CE(q^d_{y_i}, p(y|x_i^l; θ_d))
    ℓ_d ← ℓ_d / N
```

Output: The loss for each depth: ℓ_d for $d \in \{1, \dots, D\}$

Algorithm 4: SSL (Node PLs) - training step

Input: Labeled data: $\{(x_i^l, y_i) \mid i \in \{1, \dots, N\}\}$
Unlabeled data: $\{x_i^u \mid i \in \{1, \dots, M\}\}$
Depth-specific networks:
students θ_d^s, teachers θ_d^t for $d \in \{1, \dots, D\}$
Pseudo-label threshold: τ

```
// Compute the labeled loss
for d ← 1 to D do
    ℓ_d^l ← 0
    ℓ_d^u ← 0
    for i ← 1 to N do
        ℓ_d^l += CE(q_{y_i}^d, p(y|x_i^l; θ_d^s))

// Node pseudo-labels for unlabeled data
for i ← 1 to M do
    // Teacher predictions
    p(y|x_i^u) ← ProHOC({p(y|x_i^u; θ_d^t) | d ∈ {1, ..., D}})
    ŷ_i ← argmax_y [p(y|x_i^u)]

// Add pseudo-labeled data to the unlabeled loss
for 1 ← 1 to M do
    if p(ŷ_i|x_i^u) > τ then
        for d ← 1 to D do
            ℓ_d^u += CE(q_{ŷ_i}^d, p(y|x_i^u; θ_d^s))
```

Output: The loss for each depth:
$\ell_d = \ell_d^l / N + \ell_d^u / M$ for $d \in \{1, \dots, D\}$

Algorithm 5: SSL (per depth) - training step

Input: Labeled data: $\{(x_i^l, y_i) \mid i \in \{1, \dots, N\}\}$
Unlabeled data: $\{x_i^u \mid i \in \{1, \dots, M\}\}$
Depth-specific networks:
students θ_d^s, teachers θ_d^t for $d \in \{1, \dots, D\}$
Pseudo-label threshold: τ

```
for d ← 1 to D do
    ℓ_d^l ← 0
    ℓ_d^u ← 0
    // Labeled data
    for i ← 1 to N do
        ℓ_d^l += CE(q_{y_i}^d), p(y|x_i^l; θ_d^s))
    // Unlabeled data
    for i ← 1 to M do
        ŷ_i ← argmax_y [p(y|x_i^u; θ_d^t)]
        if p(ŷ_i|x_i^u; θ_d^t) > τ then
            ℓ_d^u += CE(q_{ŷ_i}^d, p(y|x_i^u; θ_d^s))
```

Output: The loss for each depth:
$\ell_d = \ell_d^l / N + \ell_d^u / M$ for $d \in \{1, \dots, D\}$

8. Learning rate and dropout for additional datasets

In Sec. 4.4 we show results from iNaturalist19, motivating our choice of learning rate and dropout. Here, we show the corresponding results for iNaturalist21-Aves and SimpleHierImageNet. Figure 10 shows results for iNaturalist21-Aves with 20 labels per class, and Fig. 11 shows results for SimplHierImagenet with 20 labels per class. Following the guideline from Sec. 4.4: choosing the largest learning rate and dropout without significant drops in ID performance, we select a learning rate of 0.01 and dropout of 0.3 for

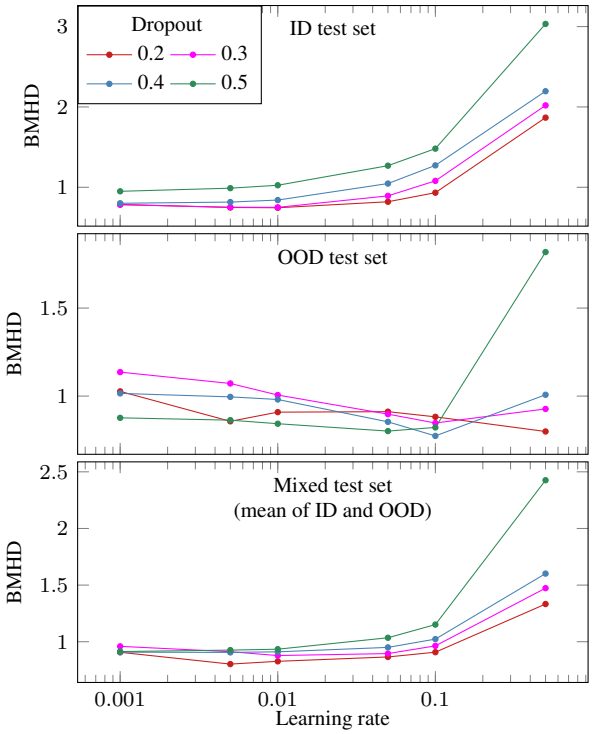

Figure 10. Varying learning rate and dropout for SemiHOC on iNaturalist21-Aves with 20 labels per class.

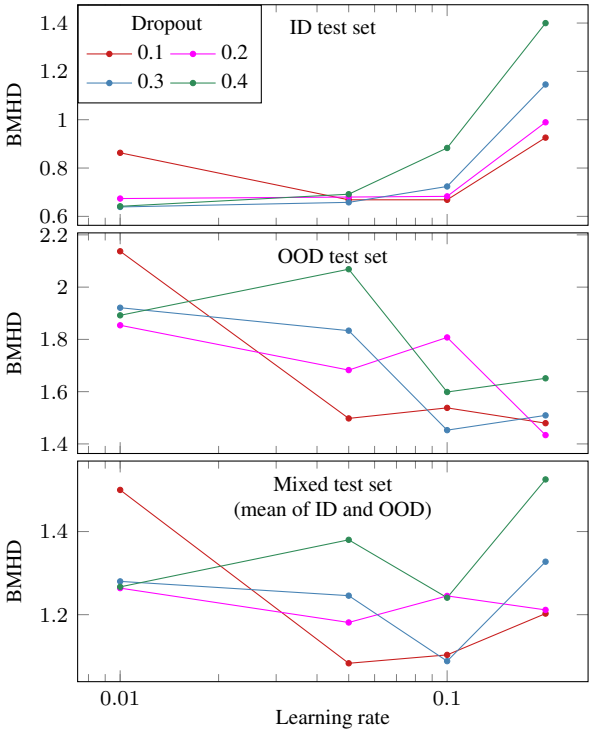

Figure 11. Varying learning rate and dropout for SemiHOC on SimpleHierImagenet with 20 labels per class.

Table 2. Statistics of our evaluated benchmark datasets.

Dataset	# ID train	# OOD train	# ID test	# OOD test	# Nodes	Depth	# ID classes	# OOD classes
SimpleHierImageNet	665,877	100,452	25,900	4,000	582	11	518	80
iNaturalist19	156,768	70,738	28,078	12,659	799	6	721	289
iNaturalist21-Aves	233,062	181,785	8,390	6,470	1,070	4	839	647

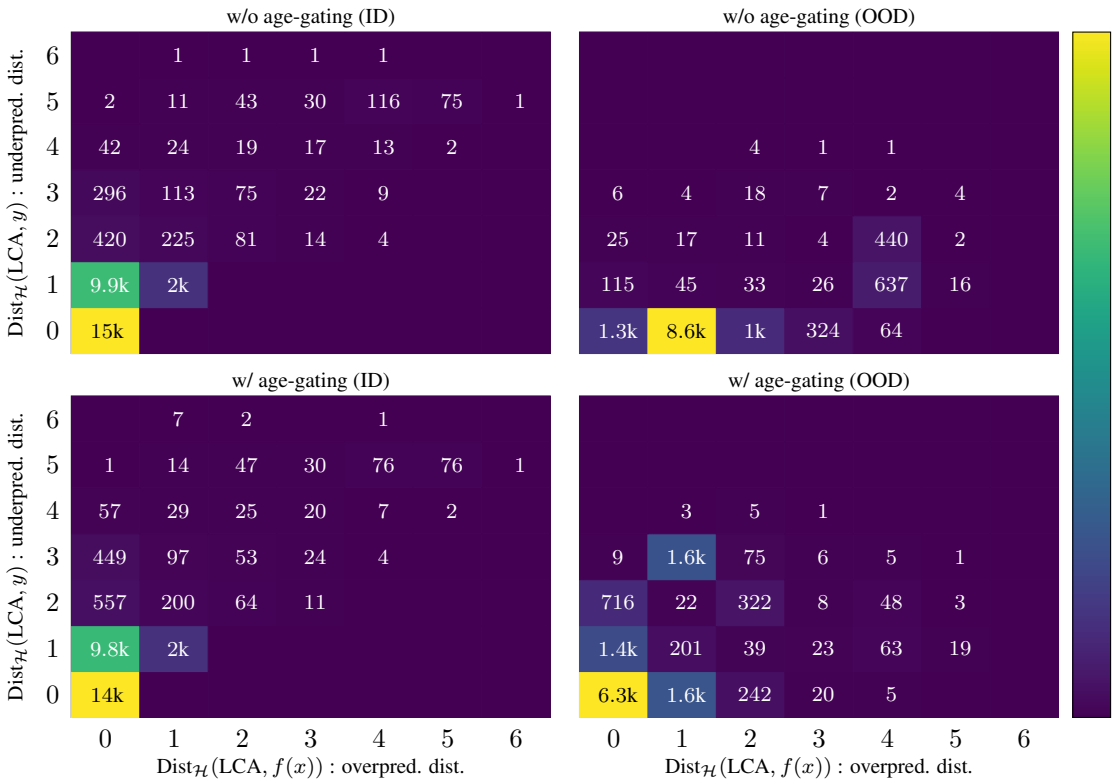

Figure 12. Hierarchical distances between predictions and ground-truths for SemiHOC with and without age-gating. The left panels show ID test data, and the right panels show OOD test data.

iNaturalist21-Aves, which is the same as for iNaturalist19. For SimpleHierImagenet, we again select dropout 0.3, but ID performance remains stable at higher learning rates, allowing us to choose 0.1.

9. Dataset details

Table 2 summarizes the benchmark datasets used in our experiments. For each dataset, we report the number of training and test samples, the number of classes, and hierarchy properties. *# Nodes* denotes the total number of nodes in the hierarchy, while *Depth* is its maximum depth. *# ID classes* corresponds to the number of leaf nodes. *# OOD classes* indicates the number of OOD classes selected from the original dataset. Note that multiple OOD classes may map to the same node in the hierarchy.

10. Distributions of hierarchical distances

To further illustrate the overconfidence issue for OOD data in semi-supervised hierarchical open-set classification, Fig. 12 shows the distribution of hierarchical distances between predictions and ground-truths for test data at the end of training for SemiHOC, both with and without age-gating. For ID data, the results are similar across the two variants. For OOD data, however, we see big differences. Without age-gating, SemiHOC predicts a majority of OOD samples too deep, with notably many samples predicted as children of the ground-truth. In contrast, with age-gating, it is most common to predict the correct node, and we see a roughly even split between over- and underpredictions. The results are from iNaturalist19 with 20 labels per class.

Figure 13. Purity and average depth of SPLs assigned to OOD data during training with and without age-gating. Results are from SimpleHierImagenet (20 labels per class).

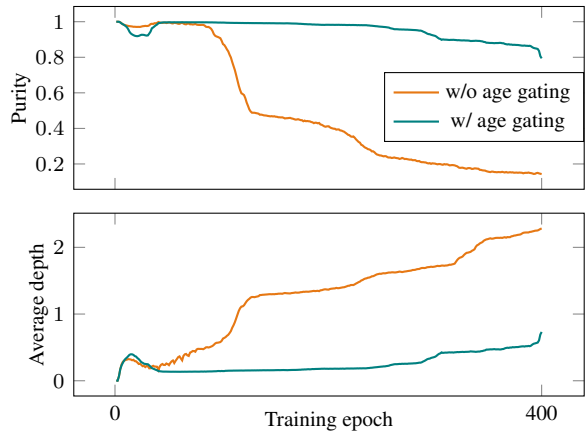

Figure 14. Purity and average depth of SPLs assigned to OOD data during training with and without age-gating. Results are from iNaturalist21-Aves (20 labels per class)

11. Generalization of age-gating across datasets

In Sec. 4.5, we analyzed the effect of age-gating on the purity and depth of subtree pseudo-labels assigned to OOD data in iNaturalist19. Here, we provide the corresponding results for SimpleHierImagenet and iNaturalist21-Aves to demonstrate that the benefits of age-gating generalize across datasets. The results are shown in Figs. 13 and 14. For both datasets, age-gating helps maintain throughout training. On SimpleHierImagenet, a challenging benchmark, the age-gated purity is lower than for the iNaturalist datasets, but the improvement over the no-age-gating baseline remains substantial.