Hierarchical Semantic Labeling With Adaptive Confidence*

Jim Davis[1], Tong Liang[1], James Enouen[1], and Roman Ilin[2]

[1] Ohio State University, Columbus OH 43210
[2] AFRL/RYAP, Wright-Patterson AFB OH 45433
{davis.1719, liang.693, enouen.8}@osu.edu, rilin325@gmail.com

Abstract. In real-world applications with semantic classification labels ('dog', 'car', 'chair', etc.), it would be advantageous to identify any unconfident classification and then determine if a less specific label could instead be reliably established. In this work, we present a hierarchical estimation and inference approach using a semantic concept tree to provide an appropriate generalized label when needed. The proposed method has several advantages, including the ability to work with any logit/softmax-based semantic label classifier, the ability to correct many misclassified labels while not introducing any new errors, and a statistical guarantee of confidence for the final labels. We additionally provide a new set of hierarchical metrics to properly evaluate the approach. Multiple synthetic and real datasets are examined to demonstrate how the framework can quickly and efficiently resolve unconfident predictions.

Keywords: Concept Tree · Classification Confidence · Semantics.

1 Introduction

Current deep learning approaches to semantic segmentation and image classification show compelling results across multiple challenging datasets (e.g., [5][2][14][11]). However, standard comparative performance metrics (accuracy, intersection-over-union, etc.) typically restrict the task and comparison to a "best-guess" for each pixel/image, though the predicted label may not be uniquely plausible. This forced-choice classification approach is obviously useful for current benchmarking purposes, but as no classifier is perfect (or ever will be) this approach can be problematic in real deployment scenarios. We believe it is important to identify an unreliable prediction and seek to find a reliable label from a set of more general options. Employing instead a top-N criteria, where credit is given if the correct answer is within the classifier's N best guesses, is still problematic as the label to choose and act upon is still unknown.

We base our approach on the natural categorical relationships of semantic labels and employ a concept tree representation that hierarchically organizes labels using IS-A relationships. For example, the initial terminal-level label 'chair'

* This work was supported by the U.S. Air Force Research Laboratory contract #GRT00044839/60056946. We acknowledge Sam Lerner for development insights.

may be deemed unconfident due to a confusion with another label (‘sofa’), occlusion, poor view-angle, etc. In this case, the parent concept ‘Seat’ would next be considered (which is composed of the related terminal descendants ‘chair’ and ‘sofa’). If ‘Seat’ is also unconfident, the tree will be traversed upward through increasingly generalized labels, such as ‘Furniture’ and ‘Object’, until a confident assessment can be found. Reaching the root node (‘Unknown’) represents a withdrawn classification, which still has practical importance since it allows one to not act on any likely false label. This approach is based on underlying label confusions, but can still address misclassifications related to occlusion, view, etc.

Consider the task of image indexing, which searches for similar images using either concept-based methods based on textual descriptions (e.g., keywords, captions, etc.) or content-based approaches using image features. One can bridge these two approaches by employing semantic segmentation to automatically derive a list of objects present in the image to use for indexing. If the forced best-guess results are used, the list of uniquely predicted labels may be corrupt due to classification errors. Hence, the image retrieval process will be sub-optimal. Alternatively, one could use a score on each pixel classification and simply discard any predictions with a low score. However, with the removal of unconfident predictions, important object labels may be missed. Instead, a method that can produce a fuller list of highly-confident, specific and generalized labels will be more descriptive and powerful.

We present a principled approach to construct a concept tree for a set of semantic labels using the natural relationships provided in a lexical database. We also provide a Bayesian method to estimate confidence throughout the concept tree using the learned posterior probability of labels given softmax values provided by a base classifier. The posterior for a given label is compared to a specified confidence threshold to determine if the label should be generalized or not. The proposed framework has multiple contributions and advantages:

- Applicable to any base classifier that outputs semantic labels and corresponding logit/softmax scores.
- Provides a statistical guarantee to meet a given level of confidence.
- Withdraws from any forced classification if a sufficiently confident label cannot be found.
- Does not produce additional incorrect classifications, and has potential to correct errors made by the base classifier.
- Enables fast and efficient label inference.
- Presents new hierarchical metrics for evaluation.

We will outline the framework and computational efficiencies, and demonstrate the effectiveness of the approach with multiple classification tasks, models, and datasets.

2 Related Work

The foundation of our work is based on a hierarchical organization of semantic labels, which we refer to as a concept tree. Other related work with such label hierarchies have been proposed. In [12], the hierarchy is used to encode prior knowledge about class similarities for knowledge transfer during training, especially for classes that have insufficient training examples. However, only terminal labels are produced in the final classification, unlike our approach which provides both terminal and non-terminal labels when needed. The top-down strategy in [8] progresses through multiple classifications (from general to specific) to reach a final terminal label. Similarly, [10] uses a tree of labels for classification by multiplying conditional probabilities along the path from the root to a particular label. For detection within a selected bounding box, the tree is traversed downward, taking the highest confidence path at every split until a threshold is reached. In these approaches, any classification error or poorly modeled conditional probability occurring early (high) in the tree will drive the decision process down a wrong branch. Our approach instead begins with the original terminal classification and softens upward through the tree only as needed until a confident label is found. In [15], pixel features are mapped to a word embedding space for label retrieval within a concept hierarchy. When used to predict terminal-only labels, the performance was lower than state-of-the-art. Their main advantage was demonstrated in zero-shot learning (not addressed in this work), where novel objects were able to be labeled to generalized concepts when their features shared enough similarity to known objects at higher levels.

Another important aspect of this work is the use of a confidence measure on classification. The softmax value associated with the best classification label is often taken as a measure of confidence. But as presented in [4], modern deep neural networks are not well calibrated, i.e., $P(l|s) \neq s$ for the softmax value s of the argmax-selected label l. Several approaches to calibrate softmax values into true probabilities have been proposed (see overview in [4]), which include histogram-based precision methods and temperature scaling of the logits (before the softmax operation). However, issues can arise in the histogram approaches when very few true and false positive predictions are available in a particular softmax bin for a class (their ratio is unstable). Also, temperature scaling is designed to calibrate either over-confident or under-confident classifications, but a class may actually be both (at different softmax ranges). Rather than attempting to calibrate with these issues, we instead take a direct Bayesian approach to measure the posterior probability of a class label given the softmax value.

3 Framework

Our proposed approach employs a semantic concept tree, which defines the hierarchical relationship between the terminal labels for a given dataset and higher, more generalized, label concepts. A measure of confidence is used to evaluate an initial label hypothesis and guide generalization upward through the tree until a sufficiently confident concept is found. The overall framework is composed of a separate estimation and inference procedure.

3.1 Concept Tree Generation

Any semantic concept tree can be employed, with more levels offering a larger range of label generalization possibilities, and the tree will necessarily be dictated by the particular application at hand. Rather than manually creating a list of ad hoc concepts and tree, we employed a non-biased, repeatable technique on each examined dataset. Our approach is based on WordNet [9], which has an internal hierarchical structure of words based on grammar usage. Other lexical relational databases or other techniques could also be used, but we chose WordNet as it is a well-established resource with existing connections to datasets and has an available Python binding (`nltk`).

To begin, one must select a particular WordNet definition for each terminal label, referred to as a *synset*, as certain labels are semantically ambiguous. For example, the noun ‘mouse’ in a dataset could refer to a rodent (n.01) or perhaps a hand-operated electronic device (n.04). Once the appropriate synset definition (n.xx) is assigned to each label, the bottom-up tree building process begins.

For each possible pairing of labels, we find their Lowest Common Subsumer (LCS), which is the deepest generalized label in WordNet where the two labels merge. For example, LCS(‘chair’, ‘sofa’) = ‘Seat’. We then select the label pair (l_i, l_j) having the deepest LCS in the WordNet hierarchy. For that label pair, we assign their LCS as their direct parent in the concept tree. The label pair is then removed from the list of labels to examine and their LCS is added to the list (if not already present). We again generate the LCS for all pairs in the new label list, and the deepest LCS is to added to the concept tree. Note that if one terminal label is a parent of another terminal label in WordNet, then we retain a separate terminal and non-terminal version of that parent label in the hierarchy. This process is repeated until only one label remains (root node), which we assign to ‘Unknown’. This automatic approach is particularly advantageous for constructing a meaningful tree with a large label set that would otherwise be manually prohibitive.

3.2 Label Confidence

A classification confidence is required at each label/node in the concept tree. As previously mentioned, modern deep learning classifiers produce softmax values s (for the argmax selected labels l) that are uncalibrated, hence $P(l|s) \neq s$. A calibration transform can be attempted with the hope of attaining $P(l|\text{Calib}(s)) = \text{Calib}(s)$, but as previously described, issues related to sample sizes and both over- and under-confidence within a class can produce undesirable results.

Instead we directly compute and evaluate $P(l|s)$, the posterior probability of a class label l given the example's softmax value s associated with that class (initially selected via argmax of the softmax vector). Using Bayes' Rule on the posterior with a two-class $\{l, \neg l\}$ context, we have

$$P(l|s) = \frac{P(s|l)P(l)}{P(s)} = \frac{P(s|l)P(l)}{P(s|l)P(l) + P(s|\neg l)P(\neg l)} \tag{1}$$

To compute the posterior distribution for a terminal node in the concept tree, we need to estimate its associated positive and negative class likelihoods and priors (to be described in the next section). With the posteriors being computed from known ground-truth class data, we do not have the previously mentioned under-sampling issue that can be present in histogram-based calibration.

3.3 Estimation Procedure

To compute the priors $P(l)$ and $P(\neg l)$ for each label l in the concept tree, class proportions found in the training set or uniform (non-informative) priors could be used. To estimate the posterior distribution, we additionally need to compute its associated positive and negative likelihood distributions. We employ a histogram-based approach, but other kernel density estimation methods could be used. For this task it is import to use validation data, which was not used to train the base classifier, as to reduce any overfitting.

For validation example x with ground-truth label l, we extract its corresponding classifier softmax value s for class l. The value of s is quantized to index into a histogram bin for this label and softmax value. As the range of softmax values is $0 \leq s \leq 1$, we can quantize s into n_b bins using $s_q = \min(\textsc{floor}(s \cdot n_b), n_b - 1)$. With this index, the *positive* histogram bin $H_l^+[s_q]$ is incremented.

Next, we need to increment the *positive* histograms for all label ancestors of l in the concept tree. Each ancestor label a will be a generalization of l and also of certain other terminal labels. For example x, we create a's softmax value using the sum of all the classifier softmax values in x for the terminal label descendants of a. This aggregated softmax value is quantized to s_q and used to increment the positive histogram bin $H_a^+[s_q]$. This process is repeated for each ancestor of l in the concept tree.

We must also increment the *negative* histograms for each label d in the set corresponding to the other terminals and non-ancestor labels/nodes of l. The associated terminals for d are identified and their corresponding classifier softmax values in x are summed, quantized, and used to index and increment the negative histogram bin $H_d^-[s_q]$.

After all validation examples have been processed, the likelihood distributions are formed by L1-normalizing the respective histograms ($P(s_q|l) = H_l^+/|H_l^+|_1$, $P(s|\neg l) = H_l^-/|H_l^-|_1$). Finally, to compute the posterior for label l, Eqn. 1 is used with $P(l|s_q) = P(s_q|l)P(l)/(P(s_q|l)P(l) + P(s_q|\neg l)P(\neg l))$. The overall estimation algorithm is shown in Alg. 1.

3.4 Inference Procedure

To make the final (confident) prediction, the inference algorithm starts with estimating the posterior probability of the base classifier's initial argmax-selected label l (at a terminal node in the concept tree) using its corresponding softmax value s. Since modern deep learning classifiers typically perform well (more often correct than incorrect), we expect the label selected via argmax to be a reasonable initial label hypothesis, to be generalized as needed. Even if the initial

Algorithm 1: Estimation

```
Organize concept tree for label set $\mathcal{L}$
Compute label priors from training data (or set equal)
foreach validation example x do
    $l$ = ground-truth($x$)                              // ground-truth label of $x$
    $sm$ = classifier_softmax($x$)                       // classifier softmax vector of $x$
    $s = sm[l]$                                          // corresponding softmax value
    $s_q$ = quantize($s, n_b$)                           // quantize into set of $n_b$ bins
    $H_l^+[s_q]$++                                       // increment positive histogram for $l$
    foreach label $a$ in ancestors($l$) do
        $\mathcal{J}$ = get_terminals($a$)               // get set of all terminals of $a$
        $s = \sum_{j \in \mathcal{J}} sm[j]$             // sum their softmax values
        $s_q$ = quantize($s, n_b$)                       // quantize into set of $n_b$ bins
        $H_a^+[s_q]$++                                   // increment positive histogram for $a$
    end
    foreach label $d$ in $\mathcal{L} \setminus (\{l\} \cup$ ancestors($l$)) do
        $\mathcal{J}$ = get_terminals($d$)               // get set of all terminals of $d$
        $s = \sum_{j \in \mathcal{J}} sm[j]$             // sum their softmax values
        $s_q$ = quantize($s, n_b$)                       // quantize into set of $n_b$ bins
        $H_d^-[s_q]$++                                   // increment negative histogram for $d$
    end
end
foreach label $l$ in $\mathcal{L}$ do
    L1-Normalize $H_l^+$ and $H_l^-$ into likelihoods
    Compute posterior $P(l|s_q), \forall q$ using Bayes with likelihoods and priors
end
```

hypothesis is incorrect, the approach may still have an opportunity to generalize to a correct label.

The softmax value s for the initial label l is quantized to s_q and used to index into the posterior $P(l|s_q)$. If the posterior is deemed unreliable (below the given confidence threshold T), there must exist other competing terminal label proposals. Therefore, we next examine the immediate parent of l with the hope of merging those competing terminal proposals. As this parent label is a generalization of the initial label and other related terminal labels, we sum all of the classifier softmax values associated with this parent. This new softmax sum is quantized and indexed into the parent label's posterior distribution. If the parent label posterior is also unconfident, we continue the process upward until a sufficiently confident label is found or the root node is reached, which by default has the label 'Unknown' with 100% confidence.

Importantly, with this method any originally *incorrect* prediction from the base classifier may be re-assigned to a valid (non-root) label that is actually an ancestor to the ground-truth label, thus *correcting* the initial error. Also, no originally *correct* prediction can be corrupted to an incorrect label as it cannot move off the upward path of the ground truth. Any initial label (either correct or incorrect) can however be removed/discarded from classification and set to 'Unknown' if no reliable label can be found. The overall inference approach is provided in Alg. 2 and is quite efficient and fast.

Algorithm 2: Inference

```
Input  : Test example $x$, confidence threshold T
Output: Confident label $l$ for $x$
$sm$ = classifier_softmax($x$)                    // classifier softmax vector of $x$
$l$ = argmax $sm$                                 // choose best class
$s = sm[l]$                                       // corresponding softmax value
$s_q$ = quantize($s, n_b$)                        // quantize into set of $n_b$ bins
Conf = $P(l|s_q)$                                 // compute posterior confidence
while Conf < T do                                 // not sufficiently confident
    $l$ = get_parent($l$)                         // use parent label
    $\mathcal{J}$ = get_terminals($l$)            // get set of all terminals of parent
    $s = \sum_{j \in \mathcal{J}} sm[j]$          // sum their softmax values
    $s_q$ = quantize($s, n_b$)                    // quantize into set of $n_b$ bins
    $Conf = P(l|s_q)$                             // compute posterior confidence
end
Assign $x \to l$                                  // final classification
```

3.5 Hierarchical Metrics

An appropriate means is needed to evaluate the method's ability to hierarchically soften/generalize, correct, and withdraw labels. There exist related metrics such as hierarchical Precision, Recall, and F-score [15]. Also, the ImageNet ILSVRC2010 competition [11][6] used the depth of WordNet's LCS between the predicted and ground-truth label. However, these metrics give partial credit for predictions that are not on the correct IS-A ancestral path of the ground truth. We instead propose a new collection of hierarchical metrics to more precisely and strictly measure various important aspects of the tree-based labeling process. In all cases, we assign no credit for a prediction off the tree path of the ground-truth. The metrics are divided into two categories related to the change in originally correct (C) and originally incorrect (IC) base predictions:

- **C-Persist** is the fraction of predictions in the set S_c (initially *correct* predictions of the base classifier) that do not change: C-PERSIST = $\frac{1}{|S_c|}\sum_{i \in S_c}$ IS-TERMINAL(l_i), where the logical IS-TERMINAL(l_i) is 1 (TRUE) if l_i is a member of the original terminal label set, else it is 0 (FALSE). Higher proportions are desired.
- **C-Withdrawn** is the fraction of initially *correct* predictions assigned to the root 'Unknown': C-WITHDRAWN = $\frac{1}{|S_c|}\sum_{i \in S_c}$ IS-ROOT(l_i), where IS-ROOT(l_i) is 1 (TRUE) if the label is assigned to the root 'Unknown' node. Lower proportions are desired.
- **C-Soften** is the fraction of originally *correct* terminal predictions that were generalized to a valid (non-root) non-terminal label: C-SOFTEN = 1 - (C-PERSIST + C-WITHDRAWN). More softened than withdrawn predictions is desired.
- **C-SoftDepth** (of C-SOFTEN) is the ratio of the tree depth between a softened label and its ground-truth, averaged over all softened labels. Larger proportions close to 1 are desired (closer to the terminal) and smaller values signify more generalization toward the root.
- **IC-Remain** is the fraction of set S_{ic} (initially *incorrect* predictions) that remain at an incorrect, non-root label: IC-REMAIN = $\frac{1}{|S_{ic}|}\sum_{i \in S_{ic}} \neg$IS-SUBSUMER$(l_i, gt_i)$ $\wedge\neg$IS-ROOT(l_i), where IS-SUBSUMER(l_i, gt_i) is 1 (TRUE) if label l_i is an ancestor of its ground-truth label gt_i. Lower proportions are desired.
- **IC-Withdrawn** is the fraction of initially *incorrect* predictions assigned to the root 'Unknown': IC-WITHDRAWN = $\frac{1}{|S_{ic}|}\sum_{i \in S_{ic}}$ IS-ROOT(l_i). Lower proportions

are desired, but as these predictions were originally incorrect, a large withdrawal is possible.

- **IC-Reform** is the fraction of initially *incorrect* predictions that are generalized to a correct, non-root label: IC-Reform = 1 - (IC-Remain + IC-Withdrawn). Larger proportions are desired.
- **IC-RefDepth** (of IC-Reform) is the ratio of the tree depth between a reformed label and its deepest possible corrected label, averaged over the set of all reformed labels $S_{ic \to c}$: IC-ReformDepth $= \frac{1}{|S_{ic \to c}|} \sum_{i \in S_{ic \to c}} \textsc{depth}(l_i)/\textsc{depth}(l_i^*)$, where l_i^* is the LCS of the original prediction and its ground-truth. Larger proportions close to 1 are desired.

4 Experiments

To evaluate the framework, we employed a synthetic classification dataset and multiple standard datasets with existing prediction models for semantic segmentation and image classification. For semantic segmentation, we examined the PASCAL VOC 2012 dataset [3] with DeepLabv3+ [2] and the ADE20K Scene Parsing dataset [1][16] with UperNet-101 [14]. The ImageNet ILSVRC 2012 dataset [11][7] with ResNet-152 [5] was used for image classification. All pre-trained prediction models are publicly available. In all experiments, we pass a single image through the model (with no additional scales or crops) to produce the base classification.

Many existing pre-trained models have been constructed using the *entire* training set, without properly using (or reporting) a validation hold-out set during the training procedure. The official validation set is commonly treated as a test set, due to unavailability of ground-truth for the actual test set. As our approach is based on the use of a validation set to properly estimate the concept tree posteriors, we therefore need to separate out a pseudo-validation and psuedo-test set from the official validation set. Rather than using a single random split, we instead employed a standard N-fold cross-validation approach. In all experiments with the real datasets, we randomly divide the validation set into N=3 partitions, then repeatedly model the posteriors on 2 partitions and test on the remaining partition. Hence, it is not straightforward to compare with other reported results (other than results of the base classifiers employed), and we therefore present average scores for the newly defined metrics.

4.1 Synthetic Experiment

We created a simple synthetic dataset to illustrate the basics of the approach. Three terminal classes (A, B, C) were each assigned 100 examples (equal priors). A tree was defined by merging terminals B and C into non-terminal D, and merging A and D into the root ('Unknown'). We set the softmax vectors for the ground-truth examples for class A all to [.8, .1, .1], for class B evenly split to {[.1, .5, .4], [.1, .4, .5]}, and for class C evenly split to {[.1, .4, .5], [.1, .5, .4]}. With the argmax as the base classification, the accuracies of A, B, and C are 100%, 50%, and 50%, respectively, simulating a strong confusion between B and C.

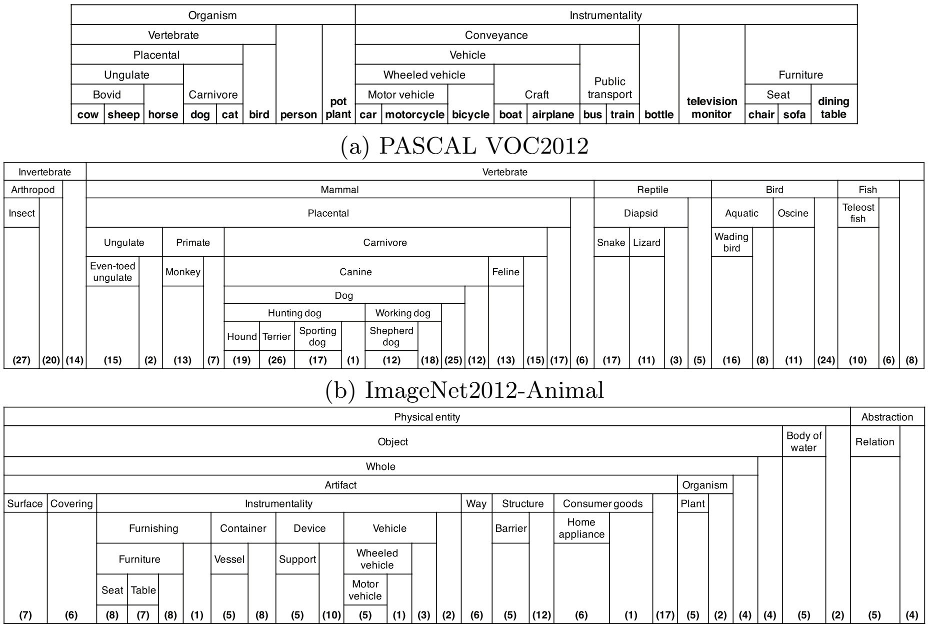
Table 1. Hierarchical metric scores for synthetic data within two confidence ranges.

	Confidence	
	≤50%	>50%
C-Persist	1	.5
C-Withdrawn	0	0
C-Soften	0	.5
C-SoftDepth	–	.5
IC-Remain	1	0
IC-Withdrawn	0	0
IC-Reform	0	1
IC-RefDepth	–	1
% Valid (¬root)	100	100
% Correct	66.7	100

Initially, the data are used to estimate the posterior distribution for each node (A, B, C, D), as outlined in Sect. 3.3. For class A, this results in a simple posterior with $P(A|s_A = .8) = 1$ and $P(A|s_A \neq .8) = 0$. For class B, we have $P(B|s_B = \{.4, .5\}) = .5$ and $P(B|s_B \neq \{.4, .5\}) = 0$. Similarly, for class C, $P(C|s_C = \{.4, .5\}) = .5$ and $P(C|s_C \neq \{.4, .5\}) = 0$. For non-terminal D, $P(D|s_D = .4 + .5 = .9) = 1$ and $P(D|s_D \neq .9) = 0$. The inference stage begins with the argmax classification results (no cross-validation is used in this experiment) which are then generalized as needed given a confidence threshold. For this example, we employed the same synthetic data for testing as to confirm the expected behavior. We report the results in Table 1.

With any confidence threshold ≤50%, the approach is equivalent to the base classification result (i.e., no classifications change). As given in the left confidence column (≤50%) in Table 1, C-Persist=1 shows all correct argmax predictions remained at their respective terminal nodes (thus C-Soften = C-Withdrawn = 0) and IC-Remain=1 denotes all incorrect predictions remained unchanged (thus IC-Reform = IC-Withdrawn = 0). Overall, every classification is a valid (non-root) label and two-thirds of them (66.7%) are correct.

With any confidence threshold >50% (right column of Table 1), *all* correct/incorrect classifications of B and C are generalized to D, while all of A remain at the correct terminal node. Hence, C-Persist = .5 (i.e., 100/(100+50+50)). Since half of B and C were originally correctly classified, but generalized to D, the value of C-Soften = .5 (i.e., (50+50)/(100+50+50)). The average softening depth C-SoftDepth = .5 reflects that those softened are halfway down the tree branch to the correct terminal. In this threshold range, all originally incorrect predictions for B and C are now at D (a correct label), hence IC-Reform = 1 (i.e., (50+50)/(50+50)). As all reformed labels are at their deepest correct label possible, IC-RefDepth = 1. There are no remaining incorrect predictions (IC-Remain=0). No labels were withdrawn to the root. In this example, the original accuracy of 66.7% (from the base classifier) was increased to 100% when the confidence is selected to be >50%.



(a) PASCAL VOC2012

(b) ImageNet2012-Animal

(c) ADE20K Scene Parse

Fig. 1. Concept trees. Default root node ('Unknown') is not shown. Values in parentheses denote number of terminals.

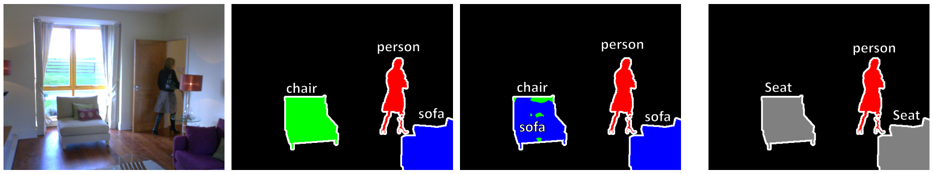
4.2 PASCAL VOC2012/DeepLabv3+

In this experiment the PASCAL VOC2012 dataset [3] and Google's DeepLabv3+ [2] pretrained model were employed. The limited VOC2012 dataset was extended with the standard trainaug labelings [13]. We ignored all 'background' and 'void' pixels as these unused pixels do not adhere to a semantic hierarchy. All estimation/inference and evaluation procedures are based on the valid 20 object labels by removing DeepLab's output logits for the unused labels and performing softmax on the remaining logits. We made the following label changes for use with WordNet: 'aeroplane' → 'airplane', 'motorbike' → 'motorcycle', 'tv/monitor' → 'television monitor', and 'potted plant' → 'pot plant'. All labels were bound to the default WordNet noun definition (n.01).

The automatically generated concept tree using the technique in Sect. 3.1 is shown in Fig. 1(a), which depicts a natural and meaningful hierarchy of labels. We computed the average hierarchical metrics using cross-validation on the resulting labelings given by our approach using priors learned from training data. A 0% confidence threshold returns the original base classification results of 97.0% accuracy (C-Persist = IC-Remain = 1). Table 2 (left) presents the average results for multiple increasing confidences. Both C-Persist and IC-Remain monotonically decrease, while C-Withdrawn and IC-Withdrawn monotonically increase (softening is based on their relative rates of change). At

Table 2. Hierarchical metric scores for multiple confidence thresholds across datasets.

	VOC2012					ImageNet-Animal					ADE20K				
	Confidence:					Confidence:					Confidence:				
	50%	75%	90%	95%	99%	50%	75%	90%	95%	99%	50%	75%	90%	95%	99%
C-Persist	.996	.979	.923	.911	.808	.853	.762	.669	.640	.620	.935	.838	.493	.126	.004
C-Withdrawn	.000	.004	.014	.017	.017	.001	.003	.007	.010	.039	.004	.013	.021	.021	.021
C-Soften	.003	.017	.063	.072	.174	.146	.235	.324	.349	.341	.061	.149	.487	.853	.975
C-SoftDepth	.703	.610	.692	.703	.480	.766	.765	.749	.728	.670	.766	.663	.767	.645	.439
IC-Remain	.924	.616	.365	.284	.205	.598	.408	.290	.267	.235	.764	.457	.325	.121	.063
IC-Withdrawn	.009	.096	.202	.227	.228	.010	.024	.041	.052	.142	.033	.062	.101	.102	.102
IC-Reform	.067	.287	.433	.489	.567	.393	.568	.669	.680	.624	.204	.482	.573	.777	.835
IC-RefDepth	.992	.968	.978	.981	.650	.976	.964	.936	.910	.842	.987	.941	.943	.890	.619
% Valid (¬root)	99.9	99.3	98.1	97.7	97.6	99.8	99.4	98.8	98.3	94.6	99.0	97.8	96.4	96.3	96.3
% Correct	97.2	98.1	98.8	99.1	99.3	91.0	93.8	95.6	95.9	96.3	84.8	90.8	93.3	97.5	98.7



(a) RGB (b) Ground-truth (c) Base prediction (d) Confident prediction

Fig. 2. VOC2012 example result. (a) RGB image. (b) Ground-truth labeling. (c) Base prediction with 'chair'/'sofa' errors, and (d) Softening and reforming of 'chair' and 'sofa' to 'Seat' at 95% confidence.

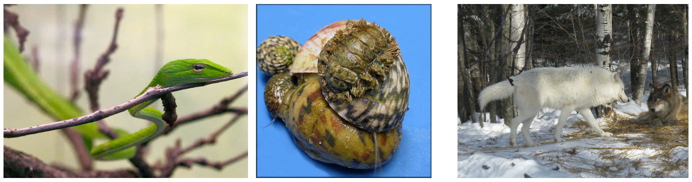
higher confidences, approximately 50% of the incorrect examples were corrected. The bottom two rows in Table 2 show that nearly all of the classifications were retained (small overall withdrawal) and that a very high percentage of them are now correct. Depending on the dataset, at 100% confidence all labels may be set to 'Unknown' (as happens here) or some non-root labels may still be selected.

An image containing the highly confusable ground-truth labels 'chair' and 'sofa' is shown in Fig. 2(a) and its ground-truth in Fig. 2(b). The DeepLab base prediction mistakenly labels most of one chair region as 'sofa' (see Fig. 2(c)). At a 95% confidence threshold our approach reasonably generalizes (softens and reforms) both chair regions to 'Seat' (see Fig. 2(d)). We note that pixels on object borders (removed in VOC2012) in most images are typically weakly predicted by a base classifier and are thus highly generalized by our approach. Furthermore, some regions may be composed of both specific and generalized labels. Additional post-processing of the final labeled image is the subject of future work.

To examine the effect of the tree itself, we employed the same tree structure but randomly shuffled the bottom 20 terminal labels. The hierarchical metrics were collected at 95% confidence and shown in the leftmost results of Table

Table 3. Additional results at 95% confidence with normal vs. randomized/shuffled tree, posterior (w/ equal priors) vs. softmax confidence, and learned vs. equal priors.

	PASCAL VOC2012		ImageNet-Animal		ADE20K	
	Normal	Shuffled	Posterior	SoftMax	Learned	Equal
C-Persist	.911	.911	.640	.178	.126	.041
C-Withdrawn	.017	.026	.010	.256	.021	.020
C-Soften	.072	.063	.349	.566	.853	.939
C-SoftDepth	.703	.557	.728	.370	.645	.653
IC-Remain	.284	.379	.267	.051	.121	.152
IC-Withdrawn	.227	.415	.052	.450	.102	.100
IC-Reform	.489	.206	.680	.499	.777	.748
IC-RefDepth	.981	.984	.910	.494	.890	.918
% Valid (¬root)	97.7	96.3	98.3	71.5	96.3	96.4
% Correct	99.1	98.8	95.9	98.9	97.5	96.9



(a) Softened to 'Snake' (b) Withdrawn (c) Reformed to 'Canine'

Fig. 3. ImageNet-Animal inference results at 95% confidence. (a) C-Soften: 'vine snake' → 'Snake'. (b) IC-Withdrawn: 'conch' → 'Unknown' (truth: 'chiton'). (c) IC-Reform: 'timber wolf' → 'Canine' (truth: 'white wolf', direct child of 'Canine').

3. As expected, C-Persist was unchanged, C/IC-Withdrawn increased, and IC-Reform significantly decreased. The use of the meaningless tree structure (for the shuffled terminals) clearly shows degraded performance.

4.3 ImageNet-Animal/ResNet-152

For this experiment, we focused on the subset of 398 animal types within the larger ImageNet ILSVRC 2012 dataset [11][7]. Microsoft's ResNet-152 [5] pre-trained model was used to give the base predictions, where all non-animal classes were removed from the output logits before computing the softmax. All animal labels were bound to their default WordNet noun definition (n.01), except for 25 labels that needed to be specified as the animal meaning (e.g., the dog 'boxer' is n.04). During the generation of the concept tree, we removed any non-terminal labels/nodes having less than 10 terminal descendants to give the more compact tree shown in Fig. 1(b).

Table 2 (middle) presents the average results using cross-validation with equal priors. The base classifier (0% confidence) has an accuracy of 85.0%, and we see that even at 50% confidence nearly all (99.8%) of the labels remain valid (non-root) and that 91% of them are correct. At higher confidences, more than 60% of the incorrect examples were reformed/corrected. A few example images are shown in Fig. 3. Notably, the withdrawn example is not clearly, or singularly, distinguished (it is a chiton on top of a mollusk) and the reformed image is relabeled more generally, yet correctly, as 'Canine' for the wolf confusion.

We also compared our posterior confidence to a baseline method directly using the softmax value as the confidence. Results at a 95% confidence threshold are presented in Table 3 (middle) and show that using softmax values alone produce much worse results, with a dramatically reduced C-Persist and IC-Reform. Testing with different thresholds on the softmax values could potentially improve these results, but would no longer be representative of a meaningful confidence measure. These results further reinforce the uncalibrated nature of softmax values, as described in [4].

4.4 ADE20K/UperNet-101

The ADE20K Scene Parsing Benchmark [1][16] contains natural images with 150 semantic labels. The particular WordNet synset definition (n.xx) was determined from the provided label information. Twenty labels were not uniquely matched and were manually assigned. The following label changes were made: 'water' → 'body of water', 'kitchen island' → 'kitchen table', and 'arcade machine' → 'slot machine' to better map to the WordNet concepts and hierarchy. The resulting concept tree with a constraint of at least 5 terminals associated with each non-terminal is shown in Fig. 1(c).

The UperNet-101 [14] pre-trained model was used as the base classifier, which produced an initial (0% confidence) accuracy of 80.3%. The hierarchical metrics with cross-validation and priors learned from training data are shown in Table 2 (right). We see a similar trend as in the previous experiments, but with much weaker scores at higher confidence ($\geq 90\%$). The small C-Persist values at high confidences can be explained by the fact that most terminal nodes have weak posteriors. For example, at 95% confidence the majority (81%) of terminal nodes cannot reach the confidence threshold (at any softmax value). In comparison, the learned posteriors for VOC2012 and ImageNet-Animal have only 10% and 2%, respectively, of terminals not able to meet the 95% threshold. Thus the UperNet-101 base classifier is not strongly confident in many cases.

We also compared the use of learned (from training data) vs. equal priors in the posterior calculation. Results at a 95% confidence threshold are presented in Table 3 (right) and show the preference for learned priors, as particularly reflected in C-Persist and IC-Reform. The difference is even more pronounced at lower confidence due to the constrained posterior values in this dataset.

5 Summary

We presented an approach that provides a useful alternative to forced-choice classification when the labels have semantically meaningful relationships that can be hierarchically organized. Instead of removing unconfident labels, the approach attempts to produce more generalized and informative labels when possible. When a label is encountered that cannot meet the required confidence at any generalization, the label is withdrawn from classification.

We outlined a bottom-up label generalization method based on WordNet to create a semantic concept tree. A Bayesian posterior confidence employing classifier softmax predictions is used to determine the level of label generalization. A separate estimation and inference process was outlined that is applicable to any base classifier producing semantic labels with logit/softmax scores. Results with multiple synthetic and real datasets demonstrated the ability to correct many errors while only mildly generalizing correctly predicted labels. The approach can be widely used with a variety of existing models and tasks to provide high confidence predictions with informative, semantic labels.

References

1. ADE20K Scene Parse dataset, http://sceneparsing.csail.mit.edu/
2. Chen, L., Zhu, Y., Papandreou, G., Schroff, F., Hartwig, A.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: ECCV (2018)
3. Everingham, M., van Gool, L., Williams, C., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2012 (VOC2012), http://host.robots.ox.ac.uk/pascal/VOC/voc2012/
4. Guo, C., Pleiss, G., Sun, Y., Weinberger, K.: On calibration of neural networks. In: ICML (2017)
5. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2015)
6. ImageNet Large Scale Visual Recognition Challenge 2010, http://image-net.org/challenges/LSVRC/2010/
7. ImageNet Large Scale Visual Recognition Challenge 2012, http://image-net.org/challenges/LSVRC/2012/
8. Liang, X., Zhou, H., Xing, E.: Dynamic-structured semantic propagation network. In: CVPR (2018)
9. Princeton University: WordNet, https://wordnet.princeton.edu/
10. Redmon, J., Farhadi, A.: YOLO9000: Better, faster, stronger. In: CVPR (2017)
11. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A., Li, F.: ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision **115**(3), 211–252 (2015)
12. Srivastava, N., Salakhutdinov, R.: Discriminative transfer learning with tree-based priors. In: NIPS (2013)
13. VOC2012 trainaug annotations, https://www.sun11.me/blog/2018/how-to-use-10582-trainaug-images-on-DeeplabV3-code/
14. Xiao, T., Liu, Y., Zhou, B., Jiang, Y., Sun, J.: Unified perceptual parsing for scene understanding. In: ECCV (2018)
15. Zhao, H., Puig, X., Zhou, B., Fidler, S., Torralba, A.: Open vocabulary scene parsing. In: ICCV (2017)
16. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ADE20K dataset. In: CVPR (2017)