LEARNING STRUCTURED REPRESENTATIONS BY EMBEDDING CLASS HIERARCHY

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing models for learning representations in supervised classification problems are permutation invariant with respect to class labels. However, structured knowledge about the classes, such as hierarchical label structures, widely exists in many real-world datasets, e.g., the ImageNet and CIFAR benchmarks. How to learn representations that can preserve such structures among the classes remains an open problem. To approach this problem, given a tree of class hierarchy, we first define a tree metric between any pair of nodes in the tree to be the length of the shortest path connecting them. We then provide a method to learn the hierarchical relationship of class labels by approximately embedding the tree metric in the Euclidean space of features. More concretely, during supervised training, we propose to use the Cophenetic Correlation Coefficient (CPCC) as a regularizer for the cross-entropy loss to correlate the tree metric of classes and the Euclidean distance in the class-conditioned representations. Our proposed regularizer is computationally lightweight and easy to implement. Empirically, we demonstrate that this approach can help to learn more interpretable representations due to the preservation of the tree metric, and leads to better generalization in-distribution as well as under sub-population shifts over multiple datasets.

1 INTRODUCTION

In supervised learning, the cross-entropy loss is often used for classification tasks. As a common practice in deep learning, in order to train a model for classification, practitioners build a linear layer over the representation to obtain the logit score of each class. A softmax transformation is then applied to convert the logits into a vector belonging to the probability simplex. As a result, we can randomly permute the representations of any classes without affecting the performance of the original classification task. However, in many real-world datasets, as we move towards fine-grained classification, labels are not independent from each other anymore: ImageNet (Deng et al., 2009) inherits label relationship from WordNet (Fellbaum, 1998), that contains both semantic and lexical connections; iNaturalist (Van Horn et al., 2017) borrows the biological taxonomy so that each image contains seven labels that reflect the morphological characteristic of the organism. Many existing works (Deng et al., 2014; Yan et al., 2014; Ristin et al., 2015; Guo et al., 2018; Chen et al., 2019) investigated how to leverage this hierarchical information for various purposes, but how to explicitly project this knowledge onto representations remains unexplored.

In this paper, we focus on the most common label relationship: tree hierarchy. As illustrated in Fig. 1b, given a tree hierarchy of classes, our goal is to learn representations in feature space such that the Euclidean distances between different class centers approximate the distances between these classes in the tree. More concretely, we shall first define a *tree metric* to be the length of the shortest path connecting two subset of classes in the tree hierarchy. Based on this tree metric, we then propose a regularizer, the cophenetic correlation coefficient (CPCC) between sequences of tree metric and Euclidean distance of the feature space, to ensure that the class-conditional representations inherit the tree structure of the classes. Different from the original cross-entropy loss with softmax activation, the proposed CPCC regularizer helps to break the symmetry of permutation invariance among the classes, and thus also improves the interpretability of the learned representations.

We show that the proposed CPCC regularizer is computationally lightweight with negligible overhead, and can be applied to a wide range of supervised learning paradigms, including standard em-

pirical risk minimization and other hierarchical objectives. For generalization, over six real-world datasets, we demonstrate that our proposed CPCC regularizer leads to improved generalization performance on some unseen tasks with sub-population shifts when there is only limited amount of labeled data, and better results on out-of-distribution detection.

2 PRELIMINARIES

In this section we first introduce the notations used throughout the paper, formulate our learning problem, and then briefly review the CPCC score to quantify the correlation of two sequences.

Notations and Setup We shall use X and Y to denote the input and target random variables, living in spaces $\mathcal{X}$ and $\mathcal{Y}$, respectively. In this work, we mainly focus on the supervised classification setting where for each input data point $x \in \mathcal{X} \subseteq \mathbb{R}^d$, there is a ground-truth label $y \in \mathcal{Y} = [k] := \{1, \ldots, k\}$, where k is the number of output classes. We let μ be the joint distribution over (X, Y) from where the data is sampled. During the learning process, the learner has access to a dataset $\mathcal{D} = \{(x_j, y_j)\}_{j=1}^n$ of size n sampled from μ.

In the context of representation learning, a learned representation $z = f_\theta(x)$ is obtained by applying a feature encoder $f_\theta : \mathcal{X} \to \mathcal{Z}$ parametrized by θ to x, where $\mathcal{Z} \subseteq \mathbb{R}^p$ denotes the feature space. Upon feature vector z, we further apply a linear predictor $g : \mathcal{Z} \to \Delta_k$, where we use Δ_k to denote the $(k-1)$-dimensional probability simplex. The cross-entropy loss is our objective function. Specifically, let $q_y \in \Delta_k$ be a one-hot vector with the y-th component being 1. The cross-entropy loss, $\ell_{\text{CE}}(\cdot, \cdot)$ between the prediction $g \circ f(x)$ and the label y is given by $\ell_{\text{CE}}(g \circ f(x), y) := -\sum_{i \in [k]} q_i \log(g(f(x))_i)$. For z, z', $\|z - z'\|_2$ denotes the Euclidean distance between them.

2.1 CLASS HIERARCHY

In classification problems, the target label $Y \in [k]$ is treated as a categorical random variable that can take k different nominal values. However, there is no particular ordering among these k categories, i.e., for different categories $i, j \in [k]$, one can only compare whether $i = j$ or not. Formally, letting $d_H(i, j) = 0$ if $i = j$ and $d_H(i, j) = 1$ otherwise defines a metric $d_H(\cdot, \cdot)$ over $\mathcal{Y}$.

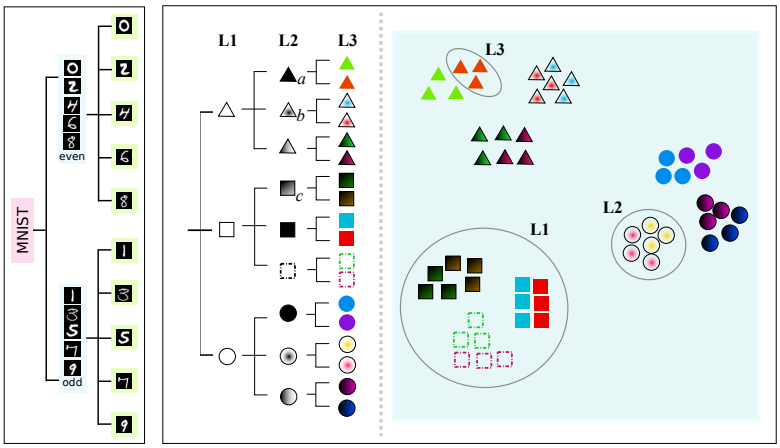
However, in many real-world applications the similarity between different classes is not binary. Consider object classification in ImageNet (Deng et al., 2009) as an example. Intuitively, one would think the distance between the classes corgi and chihuahua to be smaller than that between corgi and panda. One way to characterize this distance between different classes is through a tree of class hierarchy, also known as a dendrogram. An example is shown in Fig. 1a. Formally, let $\mathcal{T} := (V, E, d)$ be a weighted tree, where V is the set of nodes, E the set of weighted edges in $\mathcal{T}$, and $d : V \times V \to \mathbb{R}_+$ specifies the distance between nodes in the tree. Each node v_S in $\mathcal{T}$ is associated with a subset of class labels $S \subseteq [k]$, and can be recursively defined as follows:

1. For each class $i \in [k]$, there is a corresponding leaf node $v_i \in V$. Conversely, each leaf node $v_i \in V$ is identified with a single class label $i \in [k]$.
2. For some $S \subseteq [k]$, if $v_S \in V$ is not a leaf node in $\mathcal{T}$, then its children form a partition of S. In other words, if $v_{S_1}, \ldots, v_{S_c}$ are the children of v_S, then $\forall i \neq j$, $S_i \cap S_j = \varnothing$ and $\cup_{i \in [c]} S_i = S$.
3. The root node of $\mathcal{T}$ is $v_{[k]}$.

At a colloquial level, the tree $\mathcal{T}$ specifies a hierarchy of class labels that represents the structured knowledge among them. For example, as shown in Fig. 1a of the MNIST dataset, the two children of the root node correspond to the odd and even numbers, respectively. Accordingly, the distance between digits 1 and 3 is smaller than that between digits 1 and 2.

2.2 COPHENETIC CORRELATION COEFFICIENT (CPCC)

In the context of clustering, Sokal & Rohlf (1962) introduced the cophenetic correlation coefficient (CPCC) to evaluate the correspondence between two dendrograms. The CPCC is the Pearson's correlation coefficient between two sequences of pairwise distances. For a class hierarchy $\mathcal{T}$ and a node $v \in V$, the depth $dt(v)$ of v is the length of the shortest path from v to the root of $\mathcal{T}$. In the original applications of CPCC, the "dendrogrammatic " ground-truth distance $t(v_i, v_j)$ between a



(a) (b)

Figure 1: Fig. 1a: MNIST class hierarchy. The root node contains all the 10 digit classes. The two children nodes of the root node correspond to the *coarse* classes of odd and even digits, respectively. Each leaf node in this class hierarchy corresponds to a *fine* class label (digit). Fig. 1b: An example of a class hierarchy tree $\mathcal{T}$ along with a visualization of the data in the feature space. The CPCC score computes the correlation coefficient of the tree metric from $\mathcal{T}$ in the left panel and the corresponding Euclidean distance obtained from the feature space in the right panel.

pair of nodes v_i, v_j in $\mathcal{T}$ is defined as follows:

$$t(v_i, v_j) := \max\{dt(v_i), dt(v_j)\} - dt(\text{LCA}\ (v_i, v_j)),$$

where LCA (v_i, v_j) is the least common ancestor (LCA) of v_i and v_j. In Fig. 1a, the LCA of 1 and 3 is "odd" while the LCA of 1 and 0 is the root node. As an example of ground-truth distance, let us consider the class hierarchy tree $\mathcal{T}$ and clustering of classes given in Fig. 1b. In Fig. 1b, $t(\triangle_a, \triangle_b) = 1$ since they share an LCA $\triangle$ at L1, and $t(\triangle_a, \square_c) = 2$ as they go up 2 levels to meet at the the root node, which is their LCA.

Now consider a dataset $\mathcal{D}$. For a node $v_i \in \mathcal{T}$, since v_i corresponds to a subset of classes, we use $\mathcal{D}_i \subseteq \mathcal{D}$ to denote the subset of data points whose class label belongs to v_i. The pairwise distance between $\mathcal{D}_i$ and $\mathcal{D}_j$ is then defined as the Euclidean distance between the center of $\mathcal{D}_i$ and $\mathcal{D}_j$: $\rho(v_i, v_j) := \left\| \frac{1}{n_i} \sum_{x \in \mathcal{D}_i} x - \frac{1}{n_j} \sum_{x' \in \mathcal{D}_j} x' \right\|_2$, where $n_i = |\mathcal{D}_i|$ and $n_j = |\mathcal{D}_j|$ are the number of points in each cluster. Then, the CPCC score CPCC(t, ρ), between distances t and ρ is defined as:

$$\text{CPCC}(t, \rho) := \frac{\sum_{i<j} (t(v_i, v_j) - \bar{t})(\rho(v_i, v_j) - \bar{\rho})}{(\sum_{i<j} (t(v_i, v_j) - \bar{t})^2)^{1/2} (\sum_{i<j} (\rho(v_i, v_j) - \bar{\rho})^2)^{1/2}}, \tag{1}$$

where $\bar{t} := 2 \sum_{i<j} t(v_i, v_j) \,/\, k(k-1)$, $\bar{\rho} := 2 \sum_{i<j} \rho(v_i, v_j) / k(k-1)$ are the averages of all the pairwise distances for t and ρ.

3 Our Method

In this section we first define an alternative tree metric used to measure the distance between two nodes in a tree. Then, we proceed to discuss our method that uses the proposed tree metric to learn structured representations when a class hierarchy $\mathcal{T}$ is available during supervised learning.

3.1 Tree Metric

Given a tree $\mathcal{T} = (V, E, d)$ forming a class hierarchy, we formally define the *tree metric* $d_{\mathcal{T}}$ as:

Definition 3.1 (Tree Metric)**.** The tree metric $d_{\mathcal{T}}(v, v')$ for any pair of nodes $v, v' \in V$ is the weighted length of the shortest path in $\mathcal{T}$ connecting v and v'.

Proposition 3.1. For any undirected weighted graph $\mathcal{G}$, $d_{\mathcal{T}}(\cdot,\cdot)$ is a well-defined metric over $\mathcal{Y}$, if all edge weights of $\mathcal{G}$ are positive and $\mathcal{G}$ is connected.

Our main motivation to use the tree metric $d_{\mathcal{T}}$ instead of t as defined in Section 2.2 is two-folds.

First, while each edge in the class hierarchy $\mathcal{T}$ corresponds to a subset relationship, there are other kinds of structured relationships between classes that go beyond the subset relationship. For example, the benchmark dataset MetaShift (Liang & Zou, 2021) also contains a graph to describe the relationship between different subsets of classes. However, in MetaShift the weighted edges do not correspond to the subset relationship as in the case of a class hierarchy, but rather to a similarity/discrepancy measure between them. In this case, the relationship between classes corresponds to an undirected and weighted graph, where the notion of LCA does not apply any more. In fact, even in the case of a tree, the LCA is subject to change depending on which node in the tree is chosen as the root node. However, the tree metric $d_{\mathcal{T}}$ is invariant to rotations of the tree, and applies to both trees and general graphs as shown in Proposition 3.1.

Second, the definition of $t(\cdot,\cdot)$ does not account for the weights of edges in $\mathcal{T}$. This implies that all the fine-grained classes under a given super-class are the same. However, in many applications, different fine-grained classes may have different proportions or importance, for a given super-class. In these cases, the tree metric is more adapted since it also takes into account the edge weights.

Nevertheless, when $\mathcal{T}$ is an unweighted class hierarchy tree, we have the following relationship between the proposed tree metric $d_{\mathcal{T}}(\cdot,\cdot)$ and $t(\cdot,\cdot)$:

Proposition 3.2. Let $\mathcal{T}$ be an unweighted tree with a fixed root node. Then for any pair of nodes $v_i, v_j \in V$, $d_{\mathcal{T}}(v_i, v_j) = 2t(v_i, v_j) - |dt(v_i) - dt(v_j)|$.

The proof is deferred to App. A. In particular, Proposition 3.2 states that if the depths of v_i and v_j are the same in $\mathcal{T}$, then our tree metric reduces to twice of $t(v_i, v_j)$. Multiplying a variable by a constant does not affect its correlation with others. Henceforth, in what follows, we propose to use the tree metric $d_{\mathcal{T}}(\cdot,\cdot)$ in replacement of $t(\cdot,\cdot)$ to compute CPCC.

3.2 Structured Representations by Embedding the Tree Metric

Now that we have defined our tree metric $d_{\mathcal{T}}(\cdot,\cdot)$, we are interested in learning representations $f_\theta(\cdot)$, by optimizing the model parameter θ, such that the Euclidean distance between the representations of any pair of class centers $i, j \in [k]$ approximates $d_{\mathcal{T}}(v_i, v_j)$ in the tree $\mathcal{T}$. Consider a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ of size n for classification problem over Δ_k. We propose to use the CPCC between $\rho_{\mathcal{Z}}(\cdot,\cdot)$ and $d_{\mathcal{T}}(\cdot,\cdot)$ as a regularizer to the cross-entropy loss, resulting in the loss function:

$$\mathcal{L}(\mathcal{D}) = \sum_{(x,y)\in\mathcal{D}} \ell_{\text{CE}}(y, g(f_\theta(x))) - \lambda \cdot \text{CPCC}(d_{\mathcal{T}}, \rho_{\mathcal{Z}}), \tag{2}$$

where $\lambda > 0$ is the regularization strength ($\lambda = 1$ in the experiments). The Euclidean distance $\rho_{\mathcal{Z}}$ is computed in feature space. Concretely, we first apply the encoder f_θ to $\mathcal{D}$ and obtain a set of points in $\mathcal{Z} \times \mathcal{Y}$: $\mathcal{D}_{\mathcal{Z}} := \{(f_\theta(x_i), y_i)\}_{i=1}^n$. Then, we partition $\mathcal{D}_{\mathcal{Z}}$ into k subsets according to the ground-truth labels, and consider the same tree structure $\mathcal{T}$ on $\mathcal{D}_{\mathcal{Z}}$. Note the negative sign before coefficient λ in the above formulation, as we wish to maximize the CPCC score.

In practice, at each iteration during training, since stochastic optimization methods are used, we process a batch of inputs instead of the whole data set $\mathcal{D}$. For each incoming batch, we track the number of finest classes represented in the batch before any pairwise calculation. When all the inputs in a batch come from the same coarse class, the CPCC score is not well-defined due to the 0 variance of $d_{\mathcal{T}}$. This can happen when the batch size is relatively small. In such cases, we fix the value of the CPCC regularizer to 0 to avoid the numerical division by zero error.

Time Complexity of the CPCC Regularizer The computation of our CPCC regularizer is lightweight. For a feature space with p dimensions, for each training iteration, there will be at most $O(p \min(b^2, k^2))$ additional computations, where b is the batch size. Such an overhead is often negligible when compared with the computations needed to train a neural network. In Appendix B we also provide a brief discussion on the convergence of optimizing the above objective function with SGD.

3.3 THE BENEFITS OF STRUCTURED REPRESENTATIONS

In what follows, we describe two potential benefits of learning structured representations with the proposed CPCC regularizer, before providing thorough empirical validation in Section 4.

Interpretability As we briefly discussed before, one potential drawback of the representations learned through supervised learning is the lack of interpretability. Recent work (Papyan et al., 2020; Han et al., 2021) have both empirically and theoretically (under certain assumptions) shown that under the cross-entropy loss, when enough training has happened, the learned representations will have reduced variance within each class, and the set of features corresponding to different classes will converge to the so-called simplex equiangular tight frame (ETF). Yet, the vertices of the simplex ETF are symmetric (in the sense of being permutation-invariant), hence the class features do not necessarily reflect the similarities/differences between different classes, even in feature space.

By enforcing the Euclidean distances in feature space between different classes to be close to the tree metric through our CPCC regularization, we attempt to break the symmetry in learning the features. This can potentially lead to more interpretable features, as closer classes (in the sense of the tree metric) are closer to each other in feature space.

Generalization Another by-product of structured representations is potentially better generalization both in-distribution when only limited amount of labels is available, or under sub-population shifts (Santurkar et al., 2020). To see this, note that the goal of our CPCC regularizer is consistent with classification accuracy: it essentially pushes data from different classes away proportionally to their distance in the tree. Consequently, for sub-population shifts, if the hierarchy correctly captures coarse-fine relationship, future unseen fine-grained classes from the same coarse category will be further away from those under a different coarse category. This may help generalize to unseen fine-grained classes in zero or few-shot learning.

4 EXPERIMENTS

In this section, we apply our proposed method to: (i) study how using CPCC during training affects the representation learnt under various training objectives (Section 4.3), and (ii) see how the learned structured representations can improve generalization and OOD detection (Section 4.4).

4.1 DATA

We conduct our experiments on MNIST (Lecun et al., 1998), CIFAR100 (Krizhevsky, 2009), and BREEDS (Santurkar et al., 2020). By using this variety of datasets and hierarchies, we get a comprehensive overview of the usefulness of CPCC as a regularizer. See Section G for the full hierarchies.

MNIST contains handwritten digits from 0 to 9. We define odd and even digits to be two *coarse* classes (Figure 1a). The digits in the leaves are called *fine* classes below. The artificial level based on odd- and even-ness corresponds to concepts that are not visually observable. **CIFAR100** comes with a predefined hierarchy: its coarse level has 20 classes, each containing 5 fine classes (*e.g.*, beaver, dolphin, otter, seal and whale belongs to aquatic mammals). While the hierarchies are semantically meaningful, the coarse level labels are not purely defined by visual similarities. For instance, it is hard to tell the size of an animal from its image (making the coarse classes large omnivores/herbivores and mid-size mammals difficult to distinguish). **BREEDS** is a benchmark built on ImageNet (Deng et al., 2009). It contains a manually calibrated label hierarchy, based solely on shared visual characteristics. Santurkar et al. (2020) proposed four tasks: *LIVING17*, *ENTITY13*, *ENTITY30*, and *NONLIVING26*. For each, we consider the leaf nodes, which are ImageNet classes, as our fine level classes, and define the coarse levels to be their "superclasses" at different depths. We end up with trees that only contain the root node and two levels of the initial hierarchy, and ignore intermediate relationships for CPCC regularization.

New Levels Based on the coarse and fine levels in the hierarchy, we insert a *mid* level between the coarse and fine ones, as well as a *coarser* level between the coarse one and the root node. This results in classes verifying $k_{\text{coarser}} < k_{\text{coarse}} < k_{\text{mid}} < k_{\text{fine}}$. In MNIST, the mid classes are 1,3,5 (odd numbers ≤ 5), 7,9 (odd numbers > 5), 0,2,4 (even numbers ≤ 5), and 6,8 (even numbers > 5). We do not consider a coarser level (it is trivial to train on the root node). In CIFAR100, each coarse

level (containing 5 fine classes) is split into arbitrary groups of 2/3 fine classes, creating 40 classes in the mid level. 2 arbitrary coarse classes are merged into 1 coarser label, creating 10 coarser labels. Since BREEDS contains 8 non-root levels in total, and all 4 datasets' coarse levels have a depth ≥ 2, we use the original hierarchy and let the mid level be one level above the fine classes, and the coarser level be one level above the coarse classes.

Source & Target Split We split *BREEDS* into source (s) and target (t): s and t have the same coarse labels but fine classes are different, and coarser level is the same but mid level is different for s and t. Following this idea, recall we split MNIST/CIFAR's coarse levels into groups of 2 and 3. We take 60 fine classes as *CIFAR*^{s} and the rest as *CIFAR*^{t}, 6 classes as *MNIST*^{s}, 4 in *MNIST*^{t}, so source and target have different mid/fine but the same coarse/coarser labels. Due to this construction, there is only one mid class in each of *CIFAR/MNIST*$^{s/t}$'s coarse class. On the other hand, *BREEDS*'s coarse classes have many mid children. See App. G for the full hierarchies.

4.2 Baselines and Metrics

As mentioned above, we operate in a fully supervised setting. We denote our neural network as a function $h : \mathcal{X} \to \Delta_k$. $h \doteq g \circ f_\theta$ can be decomposed into a feature extractor f_θ, and a linear classifier g to which the softmax is applied. Training is performed using the following objectives, on the fine-coarse hierarchy of MNIST, CIFAR, and BREEDS, with and without CPCC as a regularization (see App. C, F for more details). Our baselines include:

- **Flat** ℓ_{CE}: training on the fine classes only without leveraging any hierarchical information.
- **Multi-task Learning**: jointly training a two-headed network to treat fine and coarse as two separate tasks. The loss function is the sum of the cross-entropies on the fine and coarse classification tasks, and we simply set the weight of the two parts to 1.
- **Curriculum Learning**: In the spirit of curriculum learning, we first train on the coarse classes using ℓ_{CE} and use y_{coarse} instead of y_{fine}. In the second step, we remove the linear classifier and fine tune a new one on the fine level labels with ℓ_{CE} as the loss function.
- **Sum Loss**: We define a hierarchical Sum Loss as $\sum \ell_{\text{CE}}(y_{\text{coarse}}, \mathbf{W}h(x)) + \ell_{\text{CE}}(y_{\text{fine}}, h(x))$, $\mathbf{W}$ is a k_1 by k_2 matrix representing the relationships in the label tree: if a fine class i belongs to a coarse class j, then $\mathbf{W}_{ji}$ is 1, otherwise the entry is set to 0.
- **HXE**: The Hierarchical Cross Entropy (Bertinetto et al., 2020) that replaces the predicted output in ℓ_{CE} with weighted hierarchical class conditional probabilities.
- **Soft**: The soft labels objective (Bertinetto et al., 2020) where labels in ℓ_{CE} are derived from a mapping function to encode class node similarity in y.
- **Quad**: The Quadruplet Zhang et al. (2016) multi-task loss which combines ℓ_{CE} with a generalized triplet loss to enforce different margins at different levels of the hierarchy.

Metrics To evaluate the representation structure learnt with the various loss functions, as well as the influence of CPCC, we use (i) silhouette scores (Rousseeuw, 1987) to measure the salience of clustering patterns at the coarse level, (ii) CPCC as a metric to measure how the whole representation structure is similar to the fine-coarse hierarchy, (iii) t-SNE (Van der Maaten & Hinton, 2008) for visualization of the learnt embeddings in 2D, and (iv) a symmetric distance matrix to evaluate the hierarchical structure. Specifically, we calculate the Euclidean distance between the mean representation vectors for each pair of fine classes. The matrix is organized in a way where fine classes from the same coarse class are grouped together, so that the coarse within-cluster distance is shown around the diagonal while other entries present coarse level between-cluster distances.

4.3 Structure of the Learnt Representations

Figure 2 show the effects of training with CPCC, which matches our expectations shown in Fig. 1b. In the distance matrix, we can see that the within-coarse cluster distance is much smaller than the between-coarse cluster distance (corresponding to diagonal 5 by 5 blocks). This fact is verified qualitatively in the t-SNE plots, where coarse groups tend to be better separated. Similar patterns are observed when CPCC is paired with the other loss functions described above. We want to point out that although other setups have a more structured representation to some extent, these are not as perfect as when paired with CPCC (see App. D for the figures).

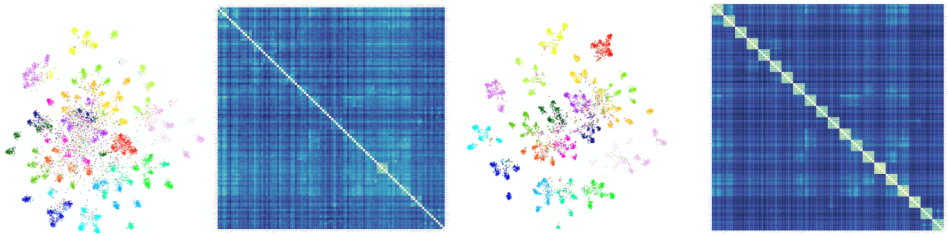


Figure 2: The matrices show the distance between fine CIFAR100 classes with and without CPCC for **Flat** (lighter color means smaller distance, same color palette used for both). The light diagonal blocks with CPCC correspond to the coarse classes. We also show t-SNE visualization of the representations (colored by coarse labels) learnt using **Flat** with and without CPCC regularization. The pattern is aligned with distance matrix. When CPCC is used, fine classes from the same coarse classes tend to be closer, and coarse classes tend to be further apart.

In Table 1, we see that the objectives leveraging the hierarchical information tend to increase the CPCC. This is particularly true of the multi-task and soft labels setting. But, directly optimizing the CPCC score still gives the largest gains, both on CPCC and silhouette scores.

4.4 Generalization on datasets with a shared hierarchy

Given a representation f_θ trained on the fine-coarse hierarchy (4.2), we want to see if structured representations help performance *in-hierarchy*, *i.e.*, on the fine and coarse classes the model was trained on; as well as *out-of-hierarchy*, *i.e.*, on new levels and/or new classes of the hierarchy.

In-hierarchy In this setting, we evaluate the models on classes and levels used during training to construct the various objectives and the tree metric (*i.e.*, the fine and coarse classes). Results can be found in the FineAcc and CoarseAcc columns of Table 1. Adding our CPCC regularizer leads to better test accuracy at both levels, across objectives and datasets, with gains sometimes exceeding 1%. According to Goyal et al. (2021), such a performance gain (especially on fine classes) is rarely observed when hierarchical information is leveraged. Overall, our findings suggest that when such information is available, using CPCC as a regularizer is beneficial.

Out-of-hierarchy Two questions naturally arise in our hierarchical setting. The first one is how well CPCC structured representations generalize to new levels of the hierarchy. To answer this question, we report in Table 1 the accuracy on the new mid and coarser levels defined in Section 4.1, and not used during training. This accuracy is obtained zero-shot, via a simple marginalization (*e.g.*, the probability of a mid class is the sum of the probabilities of all fine classes that belong to it). There too, adding the CPCC regularization results in performance gains (also see Appendix Table 4).

The second natural question is if CPCC structured representations can generalize to classes unseen in the training hierarchy. Assume a model has learned that cats and dogs are animals. Does knowing the animal concept help it understand giraffes or horses better? To explore this, we train our models on the source split of the mid and fine classes (Sec. 4.1), and evaluate their performance on the target split. We can still apply "zero-shot" transfer to coarse and coarser level via marginalization, but fine tuning is necessary to classify the new mid and fine classes: we freeze f_θ and fine tune a linear classifier g_{mid} or g_{fine} on a single image from each new target label (one-shot generalization).

Results are shown in Table 2. First, under this subpopulation shift, using CPCC still outperforms the original loss functions on coarse and coarser levels (zero-shot), which is consistent with results in Table 1. Second, in one-shot generalization to new mid levels, CPCC gives an often large advantage. Intuitively, as all fine classes are grouped together within coarse groups, if one data point is randomly selected, then other data points in the same coarse class will readily be assigned the same label. Without this structure in the representation, generalization is more difficult as all fine labels are evenly distributed. The only notable exception is *ENTITY13*, where each coarse label has many mid level children. When grouping the representations by coarse levels, it becomes difficult to recover the decision boundary of mid classes as it is close to that of other fine classes in the same

Table 1: Mean % and standard deviation over 5 seeds for various datasets, objectives and metrics, with and without CPCC (overall best in bold, best for a given objective with/without CPCC underlined). *BREEDS*'s results are on the source split. Regularizing with CPCC never hurts performance, and in most cases leads to consistent and sometimes significant improvements on all metrics.

Dataset	Objective	CPCC	Silhouette	FineAcc	MidAcc	CoarseAcc	CoarserAcc
MNIST	Flat	10.80 (1.49)	13.97 (0.72)	99.05 (0.23)	**99.38 (0.04)**	99.49 (0.08)	N/A
	FlatCPCC	**99.96 (0.01)**	**61.33 (0.42)**	**99.28 (0.08)**	**99.38 (0.03)**	**99.61 (0.03)**	N/A
CIFAR100	Flat	24.38 (0.57)	5.59 (0.02)	76.82 (0.30)	80.27 (0.35)	85.59 (0.35)	86.85 (0.27)
	FlatCPCC	84.20 (0.39)	34.40 (0.11)	77.47 (0.27)	81.30 (0.14)	86.95 (0.17)	88.17 (0.17)
	MTL	39.75 (0.33)	8.09 (0.08)	76.56 (0.20)	80.17 (0.22)	85.79 (0.20)	87.11 (0.14)
	MTLCPCC	84.88 (0.58)	31.58 (0.23)	76.90 (0.32)	80.91 (0.29)	87.11 (0.19)	88.39 (0.19)
	Curr	23.81 (0.60)	5.25 (0.11)	76.84 (0.20)	80.40 (0.17)	85.72 (0.16)	87.02 (0.18)
	CurrCPCC	**85.32 (0.51)**	34.08 (0.23)	**77.48 (0.44)**	**81.42 (0.32)**	**87.15 (0.19)**	**88.44 (0.20)**
	SumLoss	29.85 (0.63)	4.93 (0.07)	76.78 (0.20)	80.47 (0.22)	85.88 (0.25)	87.11 (0.26)
	SumLossCPCC	84.78 (0.64)	31.16 (0.13)	77.26 (0.12)	81.17 (0.18)	86.99 (0.07)	88.26 (0.02)
	HXE	25.40 (0.68)	8.31 (0.05)	76.58 (0.27)	80.17 (0.24)	85.67 (0.15)	87.02 (0.16)
	HXECPCC	85.13 (0.22)	**35.84 (0.18)**	76.57 (0.33)	80.63 (0.24)	86.48 (0.20)	87.77 (0.20)
	Soft	55.95 (0.67)	14.48 (0.11)	76.82 (0.06)	80.41 (0.07)	85.84 (0.16)	87.16 (0.07)
	SoftCPCC	85.23 (0.24)	35.80 (0.16)	77.11 (0.16)	81.02 (0.13)	86.63 (0.17)	87.93 (0.14)
	Quad	25.08 (0.26)	6.75 (0.06)	76.40 (0.28)	80.05 (0.27)	85.30 (0.11)	86.67 (0.14)
	QuadCPCC	84.65 (0.32)	34.79 (0.23)	77.10 (0.16)	80.92 (0.12)	86.78 (0.09)	88.04 (0.09)
LIVING17^s	Flat	36.74 (0.92)	8.89 (0.08)	83.66 (0.51)	89.16 (0.28)	89.72 (0.26)	92.54 (0.34)
	FlatCPCC	**93.56 (0.46)**	**48.26 (0.51)**	**84.97 (0.68)**	**90.52 (0.54)**	**91.13 (0.61)**	**93.66 (0.36)**
ENTITY13^s	Flat	34.97 (0.43)	3.12 (0.03)	82.36 (0.38)	84.52 (0.33)	90.43 (0.34)	93.38 (0.33)
	FlatCPCC	**90.98 (0.22)**	**38.68 (0.17)**	**83.37 (0.37)**	**85.53 (0.28)**	**91.50 (0.09)**	**94.22 (0.06)**
ENTITY30^s	Flat	24.29 (0.30)	3.07 (0.06)	80.81 (0.35)	82.28 (0.35)	86.00 (0.25)	89.27 (0.27)
	FlatCPCC	**73.55 (0.77)**	**29.43 (0.12)**	**82.00 (0.31)**	**83.57 (0.34)**	**87.60 (0.13)**	**90.72 (0.28)**
NONLIVING26^s	Flat	26.95 (0.31)	5.90 (0.13)	80.79 (0.36)	83.49 (0.30)	80.79 (0.36)	87.28 (0.25)
	FlatCPCC	**82.20 (0.44)**	**34.63 (0.31)**	**82.96 (0.35)**	**85.88 (0.37)**	**87.42 (0.36)**	**89.35 (0.40)**

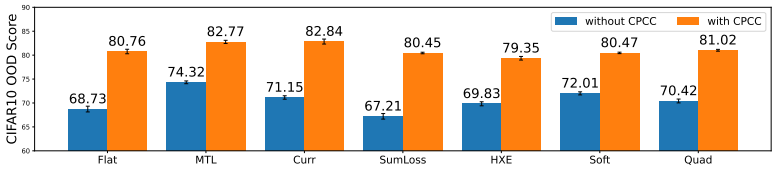


Figure 3: OOD detection score for all experiments. CPCC greatly boosts performance.

coarse class. Third, CPCC regularization is often harmful to one-shot fine level generalization due to coarse grouping: new fine classes are close together at the coarse level, making them hard to linearly separate. We do observe that other hierarchical methods have some advantage compared to the flat cross entropy. Interestingly, when coarse classes are closer to the fine leaves (*LIVING17* and *NONLIVING26* compared to *ENTITY*), imposing a coarse grouping becomes helpful, and generalization to unseen classes is improved by CPCC.

OOD Detection Finally, we explore whether OOD detection is improved by CPCC. Following Hoffmann et al. (2022), we consider CIFAR10 as OOD compared to CIFAR100 (they have disjoint fine classes), and fit a 100-class conditional multivariate Gaussian on the trained CIFAR100 embeddings. The OOD score is the probability that the model gives a higher likelihood to the true class of a random CIFAR100 image than to the maximum class likelihood of a random CIFAR10 one. Results are in Fig. 3: using CPCC significantly boosts the detection score for all objectives.

5 Related Work

Many works exploit label hierarchy, we only refer to the most related ones. The simplest label hierarchy contains only two level, coarse and fine, which can be treated as two tasks trained jointly or sequentially. The former originates from **Multi-task Learning** (MTL) where part of single network is shared for multiple heads for each task during training (Caruana, 1997; Zhao et al., 2020; Inoue et al., 2020). The latter echoes with **Curriculum Learning** (CL) (Bengio et al., 2009), where

Table 2: The superscript denotes 1- or 0-shot generalization. All models are trained on the source split s and evaluated on the target split t. s and t have different fine/mid classes but the same coarse/coarser classes. CPCC shows an advantage on mid, coarse and coarser, but not fine, levels.

Dataset	Objective	FineAcc^1	MidAcc^1	CoarseAcc^0	CoarserAcc^0
MNIST^t	Flat	**69.93 (10.46)**	53.25 (4.89)	51.84 (3.71)	N/A
	FlatCPCC	53.25 (4.89)	**55.11 (8.32)**	**58.75 (2.28)**	N/A
CIFAR100^t	Flat	28.14 (2.50)	30.66 (3.17)	42.90 (0.34)	47.37 (0.46)
	FlatCPCC	25.73 (1.05)	32.97 (6.55)	44.58 (0.17)	48.93 (0.18)
	MTL	**29.59 (1.72)**	30.36 (3.79)	42.86 (0.31)	47.34 (0.32)
	MTLCPCC	25.75 (1.61)	32.47 (5.92)	44.43 (0.44)	48.79 (0.43)
	Curr	28.77 (2.73)	30.82 (4.64)	43.88 (0.67)	48.33 (0.65)
	CurrCPCC	25.50 (1.18)	32.78 (4.47)	44.65 (0.47)	48.96 (0.47)
	SumLoss	29.31 (2.62)	30.62 (3.26)	43.15 (0.33)	47.56 (0.37)
	SumLossCPCC	26.39 (1.83)	32.40 (5.98)	**44.87 (0.37)**	**49.14 (0.39)**
	HXE	28.97 (2.81)	31.71 (5.51)	44.28 (0.43)	48.67 (0.46)
	HXECPCC	25.14 (1.84)	32.03 (5.68)	44.38 (0.34)	48.35 (0.26)
	Soft	29.35 (1.90)	32.96 (3.31)	43.99 (0.13)	48.25 (0.34)
	SoftCPCC	26.10 (1.80)	**34.10 (5.60)**	44.65 (0.70)	49.08 (0.68)
	Quad	27.89 (0.24)	31.28 (4.57)	42.73 (0.29)	47.32 (0.54)
	QuadCPCC	24.48 (1.78)	32.14 (5.57)	43.68 (0.39)	48.17 (0.42)
LIVING17^t	Flat	28.52 (3.22)	32.24 (2.60)	52.99 (0.79)	69.68 (0.38)
	FlatCPCC	**29.92 (3.40)**	**39.16 (3.30)**	**56.36 (0.69)**	**72.08 (0.43)**
ENTITY13^t	Flat	**16.03 (1.60)**	**20.66 (0.43)**	58.42 (0.33)	69.76 (0.45)
	FlatCPCC	13.06 (1.64)	18.89 (1.29)	**61.28 (0.18)**	**71.94 (0.28)**
ENTITY30^t	Flat	**21.53 (1.49)**	21.61 (1.68)	45.66 (0.20)	60.07 (0.26)
	FlatCPCC	20.79 (0.78)	**25.29 (1.32)**	**48.83 (0.12)**	**62.89 (0.39)**
NONLIVING26^t	Flat	23.50 (1.74)	25.37 (2.36)	39.31 (0.21)	53.14 (0.14)
	FlatCPCC	**24.04 (1.04)**	**27.99 (2.53)**	**42.49 (0.54)**	**56.14 (0.73)**

pretraining with a easy task will help the convergence and performance on a hard task. We define coarse level classes following Ahn et al. (2021); Peterson et al. (2018); Wang & Cottrell (2015); Stretcu et al. (2021) to emulate human learning behavior. **Sum Loss** directly modifies the loss function by marginalizing fine classes probability for coarse classes. Su & Maji (2021); Hu et al. (2018) defined a similar version to address some partially labeled tasks. Bertinetto et al. (2020) provided a thorough survey of different types of hierarchical methods, as well as label embedding methods (encoding hierarchical information into labels, the **Soft** objective), hierarchical losses (**Quad** (Zhang et al., 2016), **HXE**). None of these use a regularization method, with the exception of group overlapping lasso (Zhao et al., 2011). However, it was introduced for logistic regression, making it hard to to be applied to modern neural networks that use the penultimate layer as its representation.

The most common motivation of hierarchical models is to improve the fine-level accuracy. Interestingly, accuracy improvements are often mixed: while most works claimed to gain performance improvement, Wang & Cottrell (2015) stated that this improvement was limited, and Goyal et al. (2021) claimed most hierarchical models lead to worse performance on non-hierarchical accuracy metrics. Additionally, using coarse level labels often appears in a weakly supervised setting, where coarse classes are always available but fine class labels are only accessible for part of data, to reduce annotation cost at a finer level (Taherkhani et al., 2019; Lei et al., 2017; Ristin et al., 2015). Other works built hierarchy from dataset (Murdock et al., 2016; Li et al., 2010; Verma et al., 2012; Han et al., 2018; Zheng et al., 2017). We only name a few since they are very different from our setting where the hierarchy is defined before training.

6 Conclusion

How to include label relation into representation is an open question. In this paper, in the context of tree label hierarchies, we use the cophenetic correlation coefficient as a regularizer to embed this hierarchical relationship into representations, and outperform other baseline methods. CPCC has multiple advantages, including low time complexity, better interpretability, flexibility on any supervised learning paradigms, and it can be applied on any common label relation graphs. We also demonstrate that it leads to better generalization performance on several downstream tasks. All these benefits show that our method provides an interesting solution to this important problem.

REFERENCES

Seoyoung Ahn, Gregory J. Zelinsky, and Gary Lupyan. Use of superordinate labels yields more robust and human-like visual representations in convolutional neural networks. *Journal of Vision*, 21(13):13–13, 12 2021. ISSN 1534-7362. doi: 10.1167/jov.21.13.13. URL https://doi.org/10.1167/jov.21.13.13.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pp. 41–48, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585161. doi: 10.1145/1553374.1553380. URL https://doi.org/10.1145/1553374.1553380.

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, 2020. doi: 10.1109/CVPR42600.2020.01252.

Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311, 2018.

Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

Hao-Yun Chen, Li-Huang Tsai, Shih-Chieh Chang, Jia-Yu Pan, Yu-Ting Chen, Wei Wei, and Da-Cheng Juan. Learning with hierarchical complement objective, 2019. URL https://arxiv.org/abs/1911.07257.

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.

Jia Deng, Nan Ding, Yangqing Jia, Andrea Frome, Kevin Murphy, Samy Bengio, Yuan Li, Hartmut Neven, and Hartwig Adam. Large-scale object classification using label relation graphs. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars (eds.), *Computer Vision – ECCV 2014*, pp. 48–64, Cham, 2014. Springer International Publishing.

Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. Bradford Books, 1998.

Palash Goyal, Divya Choudhary, and Shalini Ghosh. Hierarchical class-based curriculum loss. In Zhi-Hua Zhou (ed.), *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pp. 2448–2454. International Joint Conferences on Artificial Intelligence Organization, 8 2021. doi: 10.24963/ijcai.2021/337. URL https://doi.org/10.24963/ijcai.2021/337. Main Track.

Yanming Guo, Yu Liu, Erwin M Bakker, Yuanhao Guo, and Michael S Lew. Cnn-rnn: a large-scale hierarchical image classification framework. *Multimedia tools and applications*, 77(8):10251–10271, 2018.

Xu Han, Pengfei Yu, Zhiyuan Liu, Maosong Sun, and Peng Li. Hierarchical relation extraction with coarse-to-fine grained attention. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2236–2245, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1247. URL https://aclanthology.org/D18-1247.

XY Han, Vardan Papyan, and David L Donoho. Neural collapse under mse loss: Proximity to and dynamics on the central path. In *International Conference on Learning Representations*, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. URL https://arxiv.org/abs/1512.03385.

David T Hoffmann, Nadine Behrmann, Juergen Gall, Thomas Brox, and Mehdi Noroozi. Ranking info noise contrastive estimation: Boosting contrastive learning via ranked positives. In *AAAI Conference on Artificial Intelligence*, 2022.

Peiyun Hu, Zachary C. Lipton, Anima Anandkumar, and Deva Ramanan. Active learning with partial feedback, 2018. URL https://arxiv.org/abs/1802.07427.

Matheus Inoue, Carlos Henrique Forster, and Antonio Carlos dos Santos. Semantic hierarchy-based convolutional neural networks for image classification. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2020. doi: 10.1109/IJCNN48605.2020.9207246.

Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Jason D Lee, Max Simchowitz, Michael I Jordan, and Benjamin Recht. Gradient descent only converges to minimizers. In *Conference on learning theory*, pp. 1246–1257. PMLR, 2016.

Jie Lei, Zhenyu Guo, and Yang Wang. Weakly supervised image classification with coarse and fine labels. In *2017 14th Conference on Computer and Robot Vision (CRV)*, pp. 240–247. IEEE, 2017.

Li-Jia Li, Chong Wang, Yongwhan Lim, David M. Blei, and Li Fei-Fei. Building and using a semantivisual image hierarchy. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3336–3343, 2010. doi: 10.1109/CVPR.2010.5540027.

Weixin Liang and James Zou. Metashift: A dataset of datasets for evaluating contextual distribution shifts and training conflicts. In *International Conference on Learning Representations*, 2021.

Amitangshu Mukherjee, Isha Garg, and Kaushik Roy. Encoding hierarchical information in neural networks helps in subpopulation shift, 2022. URL https://openreview.net/forum?id=hJk11f5yfy.

Calvin Murdock, Zhen Li, Howard Zhou, and Tom Duerig. Blockout: Dynamic model selection for hierarchical deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Katta G Murty and Santosh N Kabadi. Some np-complete problems in quadratic and nonlinear programming. Technical report, 1985.

Vardan Papyan, XY Han, and David L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40): 24652–24663, 2020.

Vivak Patel and Shushu Zhang. Stochastic gradient descent on nonconvex functions with general noise models. *arXiv preprint arXiv:2104.00423*, 2021.

Joshua C. Peterson, Paul Soulos, Aida Nematzadeh, and Thomas L. Griffiths. Learning hierarchical visual representations in deep neural networks using hierarchical linguistic labels, 2018. URL https://arxiv.org/abs/1805.07647.

Marko Ristin, Juergen Gall, Matthieu Guillaumin, and Luc Van Gool. From categories to subcategories: Large-scale image classification with partial class label refinement. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 231–239, 2015. doi: 10.1109/CVPR.2015.7298619.

Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987. ISSN 0377-0427. doi: https://doi.org/10.1016/0377-0427(87)90125-7. URL https://www.sciencedirect.com/science/article/pii/0377042787901257.

Shibani Santurkar, Dimitris Tsipras, and Aleksander Madry. Breeds: Benchmarks for subpopulation shift, 2020. URL https://arxiv.org/abs/2008.04859.

Robert R. Sokal and F. James Rohlf. The comparison of dendrograms by objective methods. *Taxon*, 11(2):33–40, 1962. ISSN 00400262. URL http://www.jstor.org/stable/1217208.

Otilia Stretcu, Emmanouil Antonios Platanios, Tom M. Mitchell, and Barnabás Póczos. Coarse-to-fine curriculum learning. *CoRR*, abs/2106.04072, 2021. URL https://arxiv.org/abs/2106.04072.

Jong-Chyi Su and Subhransu Maji. Semi-supervised learning with taxonomic labels, 2021. URL https://arxiv.org/abs/2111.11595.

Fariborz Taherkhani, Hadi Kazemi, Ali Dabouei, Jeremy Dawson, and Nasser M Nasrabadi. A weakly supervised fine label classifier enhanced by coarse supervision. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6459–6468, 2019.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset, 2017. URL https://arxiv.org/abs/1707.06642.

Nakul Verma, Dhruv Mahajan, Sundararajan Sellamanickam, and Vinod Nair. Learning hierarchical similarity metrics. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2280–2287, 2012. doi: 10.1109/CVPR.2012.6247938.

Panqu Wang and Garrison W. Cottrell. Basic level categorization facilitates visual object recognition, 2015. URL https://arxiv.org/abs/1511.04103.

Zhicheng Yan, Hao Zhang, Robinson Piramuthu, Vignesh Jagadeesh, Dennis DeCoste, Wei Di, and Yizhou Yu. Hd-cnn: Hierarchical deep convolutional neural network for large scale visual recognition, 2014. URL https://arxiv.org/abs/1410.0736.

Xiaofan Zhang, Feng Zhou, Yuanqing Lin, and Shaoting Zhang. Embedding label structures for fine-grained feature representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1114–1123, 2016.

Bin Zhao, Fei Li, and Eric Xing. Large-scale category structure aware image categorization. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. URL https://proceedings.neurips.cc/paper/2011/file/d5cfead94f5350c12c322b5b664544c1-Paper.pdf.

Junjie Zhao, Yuxin Peng, and Xiangteng He. Attribute hierarchy based multi-task learning for fine-grained image classification. *Neurocomputing*, 395:150–159, 2020. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2018.02.109. URL https://www.sciencedirect.com/science/article/pii/S0925231219308938.

Yu Zheng, Jianping Fan, Ji Zhang, and Xinbo Gao. Hierarchical learning of multi-task sparse metrics for large-scale image classification. *Pattern Recognition*, 67:97–109, 2017. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2017.01.029. URL https://www.sciencedirect.com/science/article/pii/S0031320317300377.

A PROOF OF PROPOSITIONS

We first restate propositions and then provide the proof.

Proposition 3.1. For any undirected weighted graph $\mathcal{G}$, $d_{\mathcal{T}}(\cdot,\cdot)$ is a well-defined metric over $\mathcal{Y}$, if all edge weights of $\mathcal{G}$ are positive and $\mathcal{G}$ is connected.

Proof. Given $v_i, v_j \in V$, if $d_{\mathcal{T}}(v_i, v_j)$ is a well-defined metric function, it means $d_{\mathcal{T}}$ must satisfy the following three properties:

1. if $v_i \neq v_j$, $d_{\mathcal{T}}(v_i, v_j) > 0$. Otherwise, $d_{\mathcal{T}}(v_i, v_j) = 0$.
2. $d_{\mathcal{T}}(v_i, v_j) = d_{\mathcal{T}}(v_j, v_i)$.
3. $\forall v_k \in V, d_{\mathcal{T}}(v_i, v_j) \leq d_{\mathcal{T}}(v_i, v_k) + d_{\mathcal{T}}(v_k, v_j)$.

By definition $d_{\mathcal{T}}(v_i, v_j)$ is the distance of the shortest path from v_i to v_j. Since $\mathcal{G}$ has positive weights, the first statement holds.

There is always a path p from v_i to v_j because $\mathcal{G}$ is connected. Since $\mathcal{G}$ is undirected, there is also a path p' from v_j to v_i by backtracking vertices on p. The sum of edge weights on p is the same as that on p'. p' must be the shortest path from v_j to v_i: if there exists a shorter path q from v_j to v_i, backtracking q will give a shorter path q' from v_i to v_j. Thus, $d_{\mathcal{T}}(v_i, v_j) = d_{\mathcal{T}}(v_j, v_i)$.

If the shortest path p from v_i to v_j contains v_k, $d_{\mathcal{T}}(v_i, v_j) = d_{\mathcal{T}}(v_i, v_k) + d_{\mathcal{T}}(v_k, v_j)$. If p does not contain v_k, if $d_{\mathcal{T}}(v_i, v_j) > d_{\mathcal{T}}(v_i, v_k) + d_{\mathcal{T}}(v_k, v_j)$, $v_i \ldots v_k \ldots v_j$ is a valid path that is shorter than p. Contradiction. ■

Proposition 3.2. Let $\mathcal{T}$ be an unweighted tree with a fixed root node. Then for any pair of nodes $v_i, v_j \in V$, $d_{\mathcal{T}}(v_i, v_j) = 2t(v_i, v_j) - |dt(v_i) - dt(v_j)|$.

Proof. Since $\mathcal{T}$ is an unweighted tree, there is a unique path connecting v_i and v_j. Let the path p be $v_i, v_1, \ldots, v_{L-1}, v_j$ of length L. Let $v_l := \text{LCA}\ (v_i, v_j)$. By definition of LCA , there exists a path from v_l to v_i and a path from v_l to v_j as well. Because the path p between v_i and v_j is unique, v_l must be on the path p as well.

Now, since v_l is the least common ancestor, the depth of v_i, v_j can be written as:

$$dt(v_i) = dt(v_l) + l, \quad dt(v_j) = dt(v_l) + L - l.$$

Hence, we have

$$\begin{aligned}
2t(v_i, v_j) - |dt(v_i) - dt(v_j)| &= 2\max\{dt(v_i), dt(v_j)\} - 2dt(\text{LCA}\ (v_i, v_j)) - |dt(v_i) - dt(v_j)| \\
&= 2\max\{dt(v_l) + l, dt(v_l) + L - l\} - 2dt(v_l) - |L - 2l| \\
&= 2\max\{l, L - l\} - |L - 2l| \\
&= \max\{l, L - l\} + \min\{l, L - l\} \\
&= L = d_{\mathcal{T}}(v_i, v_j),
\end{aligned}$$

where for the second to last equality we use the fact that $\max\{a, b\} - |a - b| = \min\{a, b\}$ for any $a, b \in \mathbb{R}$. ■

B DISCUSSION ON CONVERGENCE OF THE OBJECTIVE FUNCTION

Convergence of SGD Training with CPCC One question to ask about the CPCC regularizer is that, when coupled with the usual cross-entropy loss, can we guarantee that SGD training over the objective function in Eq. 2 will converge? Note that in general, due to the non-convexity of the loss function with deep neural networks, one cannot hope to obtain convergence to the global optimal (if exists). Absent strict assumptions on the saddle points of Eq. 2, e.g., the strict saddle point property (Lee et al., 2016) that can be hard to verify in practice, in general, it is still unclear that gradient descent will always converge to a local minimum (Murty & Kabadi, 1985). So instead, a more realistic convergence criterion that we can hope for is to show that when optimizing the

objective function in Eq. 2, SGD will converge to a stationary point. To this end, under certain regularity conditions on the data, we will show that SGD will converge to a stationary point of Eq. 2 with probability 1, by invoking a key theorem in recent advances of nonconvex optimization with SGD (Patel & Zhang, 2021, Theorem 2).

Proposition B.1 (informal)**.** For a fixed dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, assume the objective function Eq. 2 is bounded from below for every θ, and that the data is bounded, i.e., $\exists R > 0$ such that $\forall i \in [n], \|x_i\|_2 \leq R$, then under the Robbins-Monro condition of learning rate scheduling, SGD will converge to a stationary point of Eq. 2 with probability 1.

The proof of the above proposition mainly follows by verifying that the Assumptions 1-4[1] (Patel & Zhang, 2021) indeed hold in our case under the regularity condition of the data $\mathcal{D}$, so we omit it here. Essentially, in order for the regularity condition to hold, since the cross-entropy function is always bounded from below, we only need to ensure that the CPCC regularizer is bounded from below along the trajectory of θ. Note that since CPCC is a correlation score, it is bounded between $[-1, 1]$ whenever it is well-defined, i.e., the denominator is non-zero, which is the case as long as the distance $\rho_{\mathcal{Z}}(\cdot, \cdot)$ is not a constant over $\mathcal{D}$. The latter is true whenever $\mathcal{D}$ contains data points from at least two different classes and the feature map given by θ is not degenerate, i.e., at least two class mean vectors are different – a mild condition to hold in practice.

C Formulation of Objectives

- **Flat**: Train using ℓ_{CE} on fine classes only, without leveraging any hierarchical information:

$$\mathcal{L}_{\text{Flat}}(x, y_{\text{fine}}) = \sum_{(x,y)\in\mathcal{D}} \ell_{\text{CE}}(y_{\text{fine}}, h(x)). \tag{3}$$

- **Multi-task Learning**: We jointly train a two-headed network to treat fine and coarse as two separate tasks. The network then becomes $h : \mathcal{X} \to \Delta_{k_1} \times \Delta_{k_2}$ with a shared feature encoder $f_\theta : \mathcal{X} \to \mathcal{Z}$ and two classifiers $g_{\text{coarse}} : \mathcal{Z} \to \Delta_{k_1}, g_{\text{fine}} : \mathcal{Z} \to \Delta_{k_2}$. k_1 is the number of coarse classes and k_2 the number of fine classes. The loss function is the sum of the cross-entropies on the fine and coarse classification tasks, and we simply set the weight of the two parts to 1:

$$\mathcal{L}_{\text{MTL}}(x, y_{\text{coarse}}, y_{\text{fine}}) = \sum_{(x,y)\in\mathcal{D}} \ell_{\text{CE}}(y_{\text{coarse}}, g_{\text{coarse}}(f(x)) + \ell_{\text{CE}}(y_{\text{fine}}, g_{\text{fine}}(f(x))). \tag{4}$$

- **Curriculum Learning**: In the spirit of curriculum learning (Bengio et al., 2009), we first train on the coarse classes using Eq. 3 and use y_{coarse} instead of y_{fine}. In the second step, we remove the linear classifier and fine tune a new one on the fine level labels with Equation 3 as the loss function.
- **Sum Loss**: We define a hierarchical Sum Loss as

$$\mathcal{L}_{\text{SumLoss}}(x, y_{\text{coarse}}, y_{\text{fine}}) = \sum_{(x,y)\in\mathcal{D}} \ell_{\text{CE}}(y_{\text{coarse}}, \mathbf{W}h(x)) + \ell_{\text{CE}}(y_{\text{fine}}, h(x)). \tag{5}$$

  $\mathbf{W}$ is a k_1 by k_2 matrix representing the relationships in the label tree: if a fine class i belongs to a coarse class j, then $\mathbf{W}_{ji}$ is 1, otherwise the entry is set to 0. In other words, the probability of belonging to a given coarse class is simply the sum of the probabilities of its descendants (*i.e.*, fine classes) in the tree. This way, we use the information from both levels with one head only.

- **Hierarchical Cross Entropy** (Bertinetto et al., 2020) Let $h(\cdot)$ be the height of any label y_{level} in the hierarchy, $\lambda(y_{\text{level}}) = \exp(-\alpha h(y_{\text{level}}))$. In a two-level hierarchy, the HXE loss function is

$$\mathcal{L}_{\text{HXE}}(x, y_{\text{fine}}, y_{\text{coarse}}) = -\sum_{(x,y)\in\mathcal{D}} \lambda(y_{\text{fine}}) \log\Big(\frac{h(x)_{y_{\text{fine}}}}{\sum_{j\in y_{\text{coarse}}} h(x)_j}\Big) + \lambda(y_{\text{coarse}}) \log\Big(\sum_{j\in y_{\text{coarse}}} h(x)_j\Big) \tag{6}$$

  We set α as 0.4. When $\alpha = 0$, HXE is reduced to Eq. 3. As α grows up, we emphasize more on the latter term to optimize coarse level prediction.

[1]Note that the Assumption 2 of Theorem 2 in (Patel & Zhang, 2021) asks for unbiasedness of the stochastic gradient, which is not the case for the CPCC regularizer. However, as mentioned in the discussion of Theorem 2 (Patel & Zhang, 2021) as well as formally shown in Assumption 4.3(b) of (Bottou et al., 2018), this assumption could be relaxed. Fortunately, this relaxation is rather easy to account for in our case.

- **Soft Label** (Bertinetto et al., 2020) replaces the one-hot encoding of y_{fine} in Eq. 3's ℓ_{CE} by $y_{\text{fine}}^{\text{soft}}$. To leverage the hierarchy, denote $d(i, j)$ as the height of LCA(i, j) over the height of the tree. Let $i \in [k]$, the one-hot vector of y_{fine} becomes a categorically distributed vector where each component is:

$$y_{\text{soft},i}^{\text{fine}} = \frac{\exp(-\beta d(i, y_{\text{fine}}))}{\sum_{j \in [k]} \exp(-\beta d(j, y_{\text{fine}}))}, \forall i \in [k] \tag{7}$$

  We only use the two level coarse-fine hierarchy to calculate $d(i, j)$ and set $\beta = 10$. When $\beta \to \infty$, $y_{\text{fine}}^{\text{soft}}$ is the same as one-hot y_{fine}.

- **Quadruplet Loss** (Zhang et al., 2016) is a multi-task objective $\mathcal{L} = \lambda_s \mathcal{L}_{\text{Flat}} + (1 - \lambda_s)\mathcal{L}_{\text{Quad}}$ where $\mathcal{L}_{\text{Quad}}$ is:

$$\frac{1}{2N} \sum_{i=1}^{N} \max\{0, \mathscr{D}(r_i, p_i^+) - \mathscr{D}(r_i, p_i^-) + m_1 - m_2\} + \max\{0, \mathscr{D}(r_i, p_i^-) - \mathscr{D}(r_i, n_i) + m_2\} \tag{8}$$

  In Eq. 8, N is the number of valid quadruplets in the mini-batch, $\mathscr{D}(\cdot, \cdot)$ is the squared Euclidean distance of l_2 normalized features. For each incoming mini-batch, we set all images as anchor r_i, p_i^+ has the same fine label with r_i, p_i^- has the same coarse bot not fine label with r_i, and n_i and r_i have different coarse labels. We set $\lambda_s = 0.8, m_1 = 0.25, m_2 = 0.15$ in our experiments. Margin m_1 is set to be bigger than m_2 to ensure same fine labels are grouped together, then fine labels within one coarse class are close, and finally classes not in the same coarse class are far away, i.e., $\mathcal{D}(r_i, p_i^+) + m_1 < \mathcal{D}(r_i, p_i^-) + m_2 < \mathcal{D}(r_i, n_i)$. Once we found $\mathcal{D}(r_i, p_i^+)$, we pick the hardest $\mathcal{D}(r_i, p_i^-)$ and then $\mathcal{D}(r_i, n_i)$ to accelerate convergence.

D Additional Figures

We include t-SNE visualization of representations and distance matrices for the remaining objectives (Multi-task, Curriculum, Sum Loss) of Table 1 in this section (Figures 5, 6 and 7 for the t-SNE, and Figures 8, 9 and 10 for the matrices). We also include the distance matrix for using CPCC on three layers in Figure 4, showing that CPCC can generalize to deeper hierarchies without sacrificing too much classification accuracy.

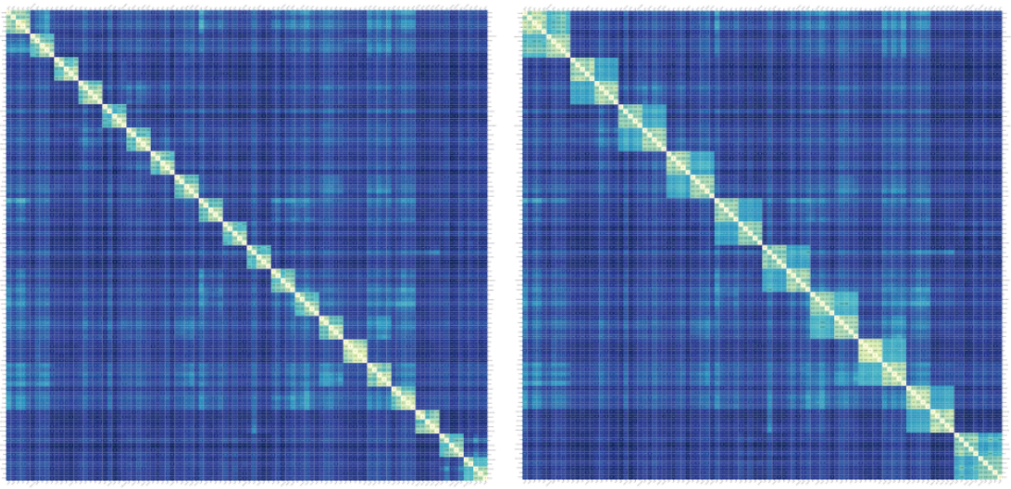


Figure 4: CPCC for three layers for **Flat**. *Left* matrix uses CPCC on fine/mid/coarse (FineAcc^0 = 77.24, CoarseAcc^0 = 86.71). *Right* matrix uses CPCC on fine/coarse/coarser (FineAcc^0 = 76.95, CoarseAcc^0 = 86.58). Both of them correctly encoded information of three levels, by either partitioning the 5 by 5 coarse diagonal blocks into a 2×2 on the left top and 3×3 on the right bottom, or grouping neighboring 5 by 5 coarse blocks. Lighter color means smaller distance.

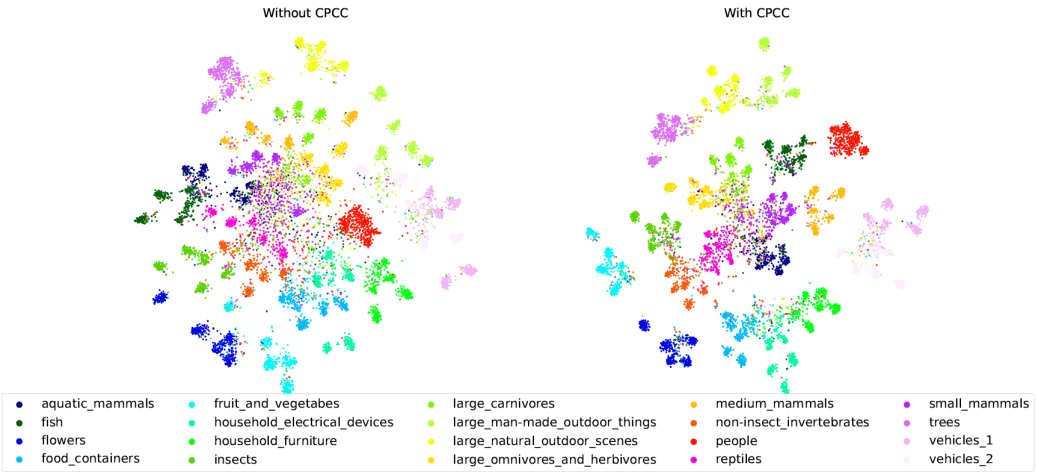


Figure 5: t-SNE visualization of the representations learnt by **Multi-task** with and without CPCC regularization.

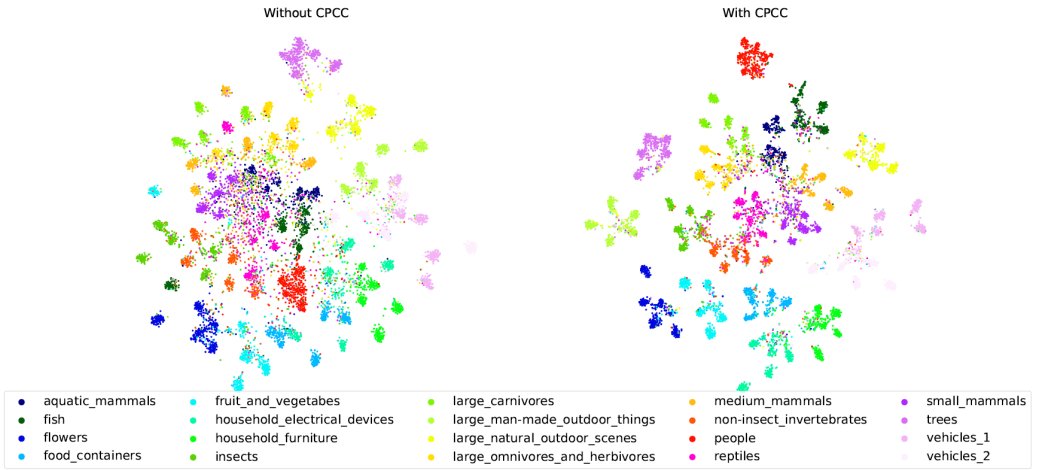


Figure 6: t-SNE visualization of the representations learnt by **Curriculum** with and without CPCC regularization.

E ADDITIONAL EXPERIMENTS

In this section, we include remaining experiments of Table 1 to Table 4 and Table 2 to Table 5. Furthermore, we discuss an extra setting where CPCC can be applied.

CPCC for sub-population shift Santurkar et al. (2020) proposed BREEDS benchmark for sub-population shift. Compared to 4.3, the main difference between two tasks is that BREEDS was only trained on coarse level, while 4.3 is trained on fine level for all objectives without CPCC. From the definition of sub-population shift, if we train on the train split of source dataset, there will be a big drop of performance of test performance on source set and target set. Since models are always trained on *coarse labels*, fine tuning is unnecessary. In Table 3, we show the performance gain of using CPCC on the Flat method. Since BREEDS contains more than two levels of hierarchy, we present the result of using CPCC on coarse-fine classes, coarse-coarser classes as examples. The result shows CPCC alleviates the sub-population shift problem by using multiple labels in hierarchy, as shown in Mukherjee et al. (2022) who used other methods.

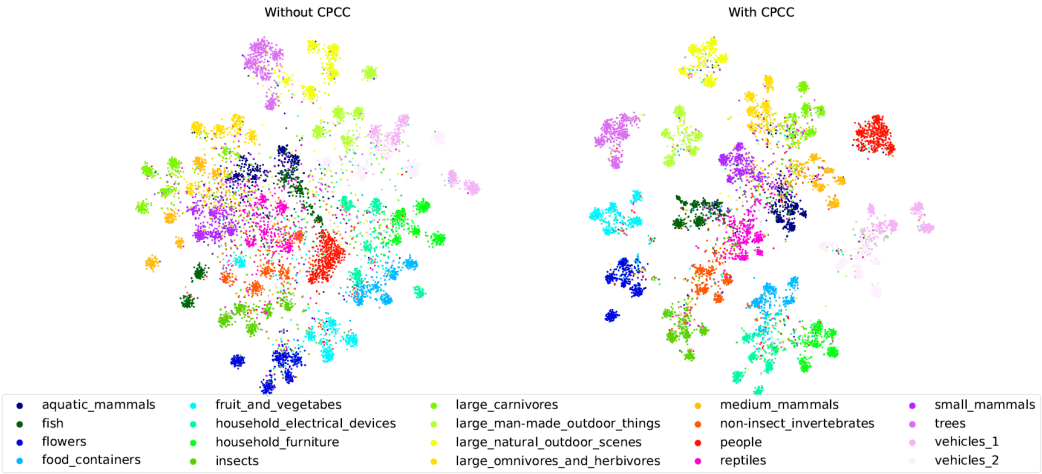


Figure 7: t-SNE visualization of the representations learnt by **Sum Loss** with and without CPCC regularization.

Objective	*LIVING17*	*ENTITY13*	*ENTITY30*	*NONLIVING26*
Flat (coarse)	56.01 (0.71)	60.49 (0.35)	47.72 (0.27)	40.25 (0.66)
Flat + CPCC (coarse, fine)	**57.74 (0.34)**	**61.32 (0.18)**	**48.99 (0.29)**	**42.31 (0.87)**
Flat + CPCC (coarser, coarse)	57.20 (0.90)	60.84 (0.25)	48.45 (0.67)	41.94 (0.90)

Table 3: BREEDS's performance for sub-population shift. Each entry is the target validation accuracy of coarse level classes. Using CPCC consistently outperforms naive Flat.

F Implementation Details

In this section, we describe the training details for our experiments. We want to highlight again that we use these parameters to show CPCC's advantage over other methods in controlled settings, but not to reproduce SOTA results.

All experiments were trained on NVIDIA RTX A6000. Experiments in Sections 4.3 and 4.4 have the same training setting, except for some optimizer setting differences. For a fair comparison, we searched hyperparameters for 4.4 to achieve the best performance for all Flat methods, and apply the same setting to other objectives with or without CPCC.

Compared with other approaches, curriculum learning in 4.3 contains two stages. Empirically, we find that a few first stage training iterations on coarse labels give a better initialization of the second stage on fine labels. The epochs of the first stage do not have a huge impact on the final performance, but less epochs on the second stage can harm the result. Therefore, we train more epochs on curriculum learning settings so that the fine accuracy for all without CPCC objectives are similar. We spend approximately the same number of epochs on fine labels for all experiments, and add some extra iterations for curriculum learning to train on coarse labels. Training epochs are the same for generalization experiments, given the fact that all representations have similar fine and coarse accuracy.

MNIST The feature encoder is composed of these layers sequentially: a 5×5 convolutional layer, 2×2 max pooling layer, a 5×5 convolutional layer, 2×2 max pooling layer, fully connected layer with output 120 units, and a fully connected layer with output 50 units. We set batch size as 256, epochs as 70, learning rate start from 0.05, divided by 5 every 25 steps, with momentum of 0.9, weight decay 5×10^{-4} of SGD optimizer. No data augmentation is applied. For curriculum learning, we train 18 epochs on coarse labels and 72 epochs on fine labels. We train one-shot transfer on mid level for 70 epochs, learning rate from 0.005 divided by 10 every 25 steps; fine level 70 epochs learning rate from 0.1 divided by 10 every 25 steps.

CIFAR100 We train ResNet18 from scratch. Following the original ResNet (He et al., 2015) paper, the first convolution layer has a 3×3 kernel size, and the max pooling layer is not included.

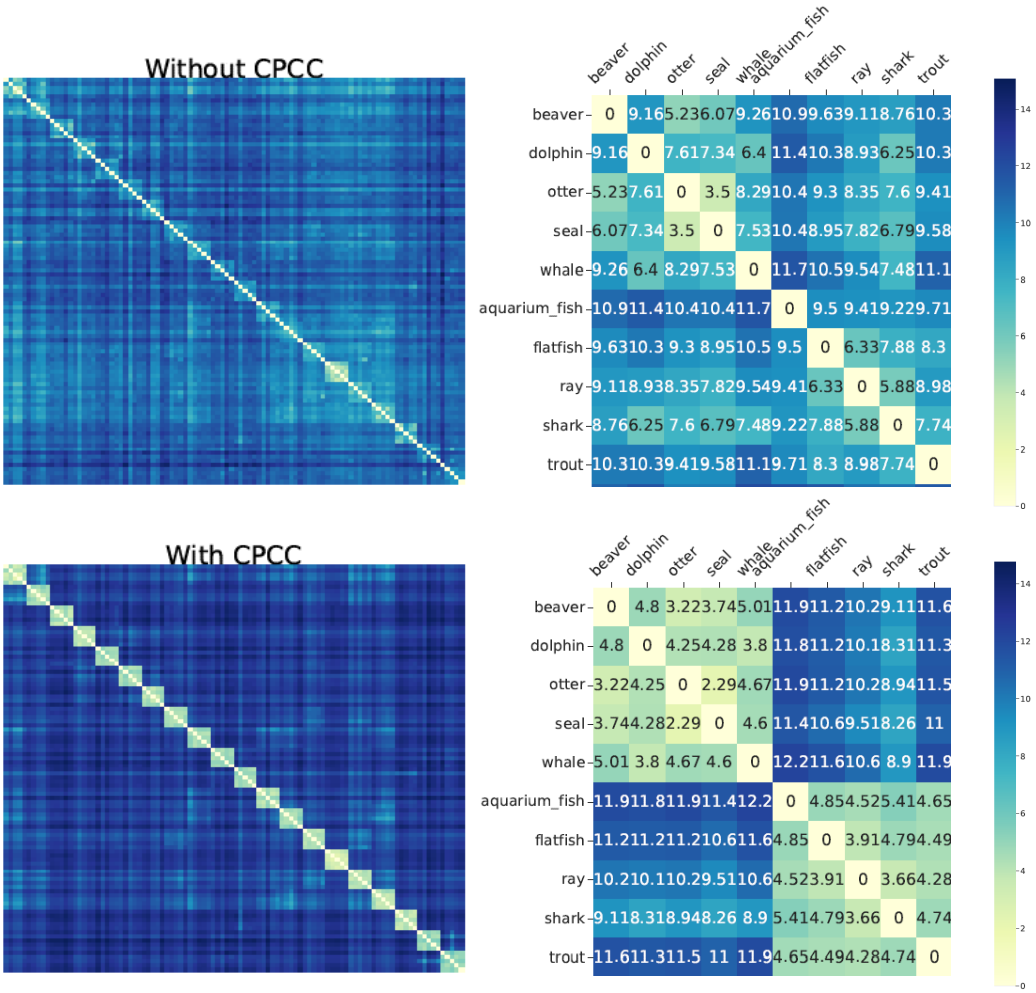


Figure 8: Distance between fine CIFAR100 classes with and without CPCC for **Multi-task**. Two matrices on the right are zoomed in to show the top left corner of the corresponding distance matrix on the left.

We set batch size as 128, epochs as 200, learning rate starts from 0.1, divided by 5 every 60 steps, with momentum of 0.9, weight decay 5×10^{-4} of SGD optimizer. All images are normalized by CIFAR100's mean and standard deviation. The following data augmentations are applied on train set sequentially: images are zero padded by 4 pixels and cropped into original size; images are flipped horizontally by the chance of 50%; images are randomly rotated by -15 to 15 degrees. For curriculum learning, we train 50 epochs on coarse labels and 200 epochs on fine labels. We train one-shot transfer on mid and fine level for 100 epochs, learning rate from 0.05 divided by 10 every 60 steps.

BREEDS We train ResNet18 from scratch. The first convolution layer has a 7×7 kernel size. We follow BREEDS training setting: we set batch size as 128, initial learning rate as 0.1, weight decay 10^{-4} of SGD optimizer. On *ENTITY13* and *ENTITY30*, we train 300 epochs, learning rate divided by 10 every 100 steps; on *LIVING17* and *NONLIVING26*, we train 450 epochs, learning rate divided by 10 every 150 steps. For curriculum learning, on *ENTITY13* and *ENTITY30*, we train 75 epochs on coarse labels and 300 epochs on fine labels; on *LIVING17* and *NONLIVING26*, we train 108 epochs on coarse labels and 432 epochs on fine labels. Train set is augmented by a RandomResizedCrop into 224×224, flipped horizontally by the chance of 50

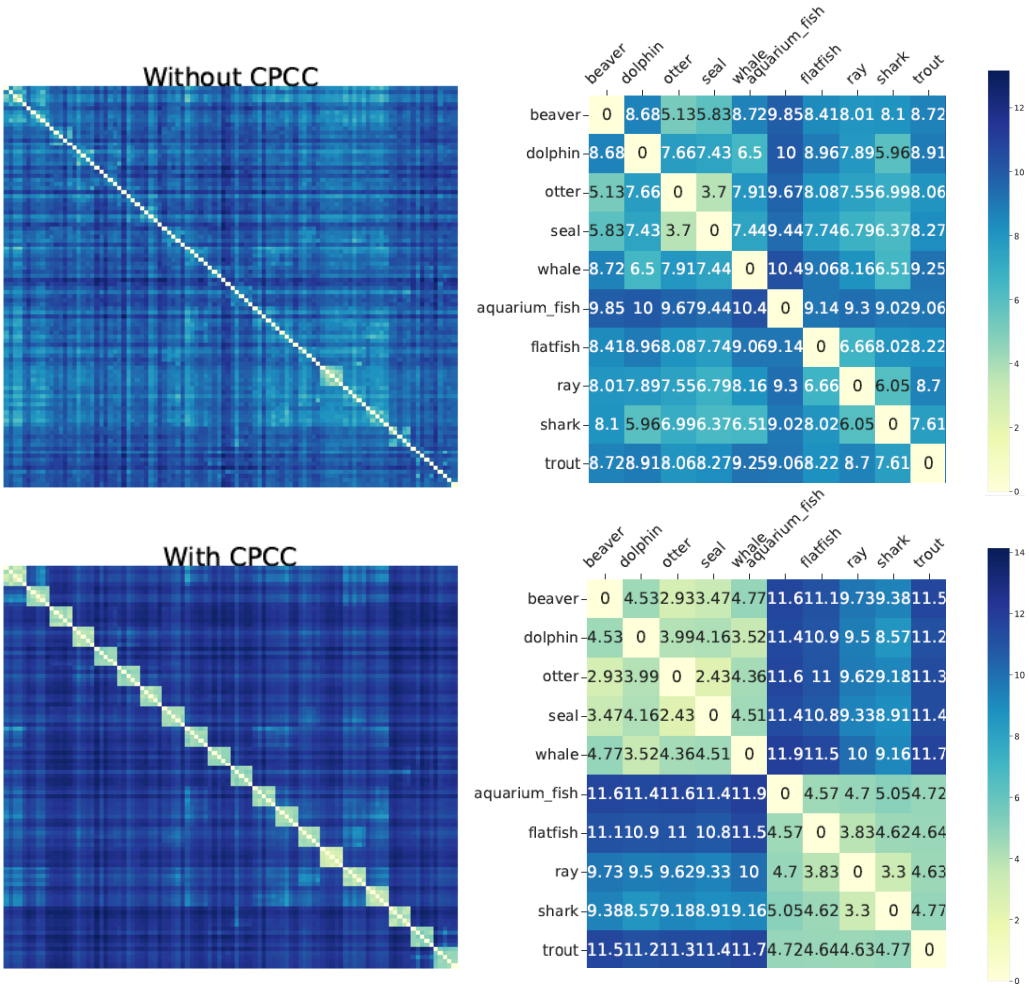


Figure 9: Distance between fine CIFAR100 classes with and without CPCC for **Curriculum**.

%, and randomly change the brightness, contrast, saturation by 0.9 to 1.1 (ColorJitter). Test set is first resized into 256×256 and CenterCropped into 224×224.

On *LIVING17*, we train one-shot transfer on mid level for 150 epochs, learning rate 0.01; fine level 150 epochs learning rate from 0.1 divided by 10 at epoch 100. On *ENTITY13*, we train one-shot transfer on mid level for 100 epochs, learning rate starting from 0.1 divided by 10 every 50 steps; fine level 200 epochs learning rate from 0.05 divided by 10 at epoch 150. On *ENTITY30*, we train one-shot transfer on mid level for 100 epochs, learning rate 0.1 divided by 10 every 50 steps; fine level 150 epochs learning rate from 0.1 divided by 10 at epoch 100. On *NONLIVING26*, we train one-shot transfer on mid level for 100 epochs, learning rate 0.1 divided by 10 every 50 steps; fine level 150 epochs learning rate from 0.1 divided by 10 at epoch 100.

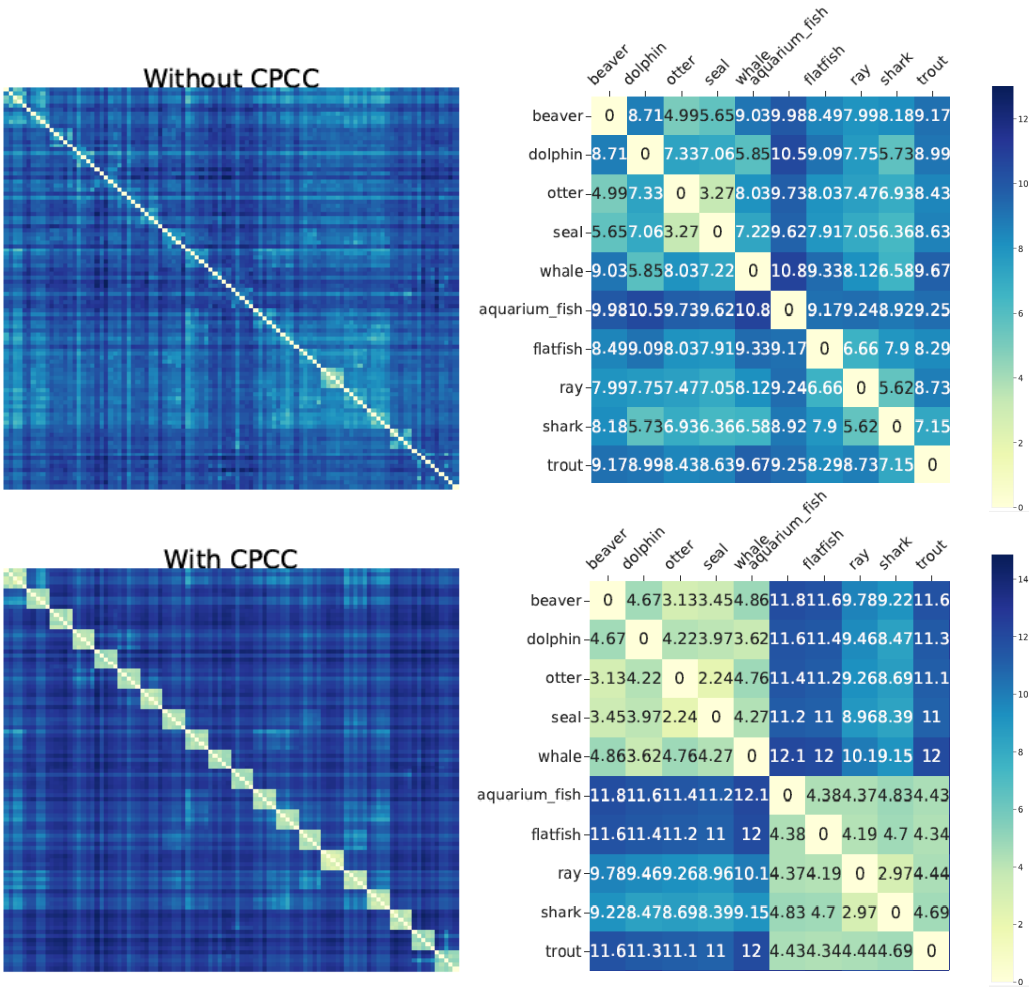


Figure 10: Distance between fine CIFAR100 classes with and without CPCC for **Sum Loss**.

G Dataset Hierarchies

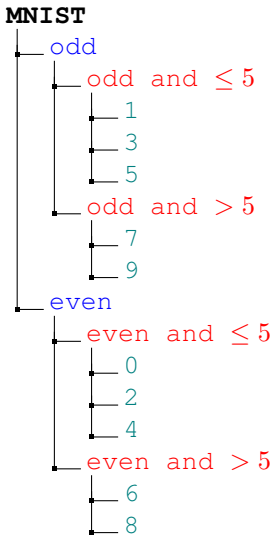
We visualize 4 levels of hierarchy (coarser, coarse, mid, fine) of MNIST, CIFAR, and BREEDS^s in Fig 11, and provide more details for Section 4.1 and 4.4 about how we constructed the hierarchy based on original labels. We can notice in BREEDS hierarchies, there are some repetitive labels[2]. One case is that the parent node is a dummy node which shares the same name as its child. For example, dummy 52 and n02773037 (WordNet ID) are both called "bags". Santurkar et al. (2020) inserted dummy nodes into the ImageNet hierarchy to adjust the granularity of each level. Since we create mid labels by backtracking the fine classes by 1 level, the other case of repetitive labels happens when a coarse label coincides with the mid label (*i.e.*, if the coarse label is exactly one level higher than the fine label). Unlike on MNIST and CIFAR, we do not want to create some artificial new classes on BREEDS. Therefore, we can also treat repetitive nodes as dummy nodes as in the first case, which only affects mid class results.

[2]All label names of BREEDS can be found in `https://github.com/MadryLab/BREEDS-Benchmarks/blob/master/imagenet_class_hierarchy/modified/node_names.txt`

Table 4: Remaining results for Table 1. Since the MNIST's hierarchy is artificial, CPCC sometimes leads to slightly worse performance (though well within the standard deviation). On BREEDS and CIFAR100, CPCC is able to use the extra visually meaningful hierarchical information to provide gain across all metrics and objectives. Also, we observe that curriculum learning works well on most BREEDS setting and gives the highest CPCC and silhouette values.

Dataset	Objective	CPCC	silhouette	FineAcc	MidAcc	CoarseAcc	CoarserAcc
MNIST	MTL	42.65 (2.96)	20.87 (1.02)	99.21 (0.05)	**99.41 (0.02)**	**99.62 (0.02)**	N/A
	MTLCPCC	**99.96 (0.01)**	60.60 (0.50)	**99.26 (0.05)**	99.34 (0.07)	99.55 (0.04)	
	Curr	31.22 (4.99)	19.09 (1.17)	99.20 (0.12)	99.37 (0.08)	99.59 (0.06)	
	CurrCPCC	**99.96 (0.01)**	**60.90 (0.31)**	99.18 (0.10)	99.36 (0.05)	**99.62 (0.03)**	
	SumLoss	23.76 (4.10)	17.05 (0.62)	99.11 (0.08)	99.36 (0.06)	99.60 (0.06)	
	SumLossCPCC	**99.96 (0.01)**	60.07 (0.75)	99.10 (0.12)	99.30 (0.09)	99.56 (0.06)	
LIVING17^s	MTL	45.83 (0.78)	11.66 (0.09)	83.87 (0.28)	89.60 (0.23)	90.29 (0.27)	93.15 (0.37)
	MTLCPCC	90.71 (4.28)	45.05 (0.54)	84.77 (0.59)	90.84 (0.39)	91.53 (0.30)	94.07 (0.33)
	Curr	40.19 (0.71)	10.95 (0.12)	83.87 (0.40)	89.53 (0.54)	90.10 (0.49)	93.01 (0.52)
	CurrCPCC	**93.41 (0.34)**	**51.36 (0.46)**	**85.21 (0.42)**	90.91 (0.53)	91.59 (0.65)	94.07 (0.59)
	SumLoss	39.02 (1.10)	8.95 (0.26)	83.49 (0.63)	89.24 (0.46)	89.92 (0.31)	92.74 (0.19)
	SumLossCPCC	93.03 (0.46)	44.36 (0.28)	85.05 (0.23)	**91.07 (0.21)**	**91.79 (0.25)**	**94.56 (0.19)**
ENTITY13^s	MTL	43.55 (0.71)	5.19 (0.11)	82.47 (0.28)	84.80 (0.04)	91.13 (0.10)	93.97 (0.15)
	MTLCPCC	90.73 (0.34)	36.62 (0.27)	82.83 (0.23)	85.06 (0.21)	91.51 (0.24)	94.21 (0.17)
	Curr	37.03 (0.51)	3.98 (0.05)	83.41 (0.29)	85.39 (0.31)	91.15 (0.15)	93.98 (0.13)
	CurrCPCC	**92.38 (0.25)**	**41.06 (0.15)**	**83.86 (0.25)**	**86.12 (0.18)**	**91.90 (0.20)**	**94.55 (0.19)**
	SumLoss	37.99 (0.26)	3.51 (0.02)	82.50 (0.23)	84.86 (0.16)	90.83 (0.25)	93.68 (0.18)
	SumLossCPCC	89.66 (0.30)	35.08 (0.23)	82.97 (0.43)	85.26 (0.32)	91.56 (0.10)	94.24 (0.08)
ENTITY30^s	MTL	30.55 (0.26)	5.25 (0.12)	81.31 (0.32)	82.81 (0.19)	86.95 (0.15)	90.12 (0.15)
	MTLCPCC	72.00 (0.52)	25.92 (0.19)	81.90 (0.17)	83.58 (0.20)	87.66 (0.15)	90.72 (0.15)
	Curr	27.09 (0.20)	4.55 (0.05)	82.19 (0.45)	83.59 (0.42)	87.52 (0.25)	90.54 (0.31)
	CurrCPCC	**77.80 (0.50)**	**34.08 (0.19)**	**82.86 (0.20)**	**84.31 (0.20)**	**88.55 (0.09)**	**91.23 (0.14)**
	SumLoss	26.05 (0.29)	3.18 (0.09)	81.11 (0.17)	82.59 (0.21)	86.62 (0.14)	89.71 (0.10)
	SumLossCPCC	66.10 (0.63)	21.38 (0.20)	81.76 (0.26)	83.28 (0.26)	87.63 (0.25)	90.68 (0.18)
NONLIVING26^s	MTL	30.38 (0.73)	7.36 (0.10)	82.06 (0.24)	84.91 (0.15)	86.42 (0.22)	88.46 (0.31)
	MTLCPCC	79.49 (0.66)	30.55 (0.22)	82.84 (0.61)	85.91 (0.42)	87.60 (0.39)	89.45 (0.45)
	Curr	28.81 (0.41)	7.68 (0.08)	81.82 (0.65)	84.72 (0.69)	86.23 (0.73)	88.35 (0.54)
	CurrCPCC	**81.70 (0.71)**	**36.92 (0.25)**	**83.54 (0.28)**	**86.45 (0.30)**	**88.06 (0.29)**	**89.83 (0.32)**
	SumLoss	28.57 (0.69)	5.65 (0.15)	81.58 (0.97)	84.56 (0.71)	86.28 (0.63)	88.48 (0.56)
	SumLossCPCC	79.04 (0.95)	29.14 (0.36)	82.60 (0.51)	85.82 (0.54)	87.42 (0.41)	89.27 (0.51)

Figure 11: Hierarchies for all datasets.



(a) MNIST hierarchy.

Table 5: Remaining results for Table 2. As in Table 4, MNIST's results occasionally (SumLoss-CPCC MidAcc, Curriculum CoarseAcc) show CPCC is worse than original objective due to artificial hierarchy, but on BREEDS where we have a more reasonable visual hierarchy, the conclusion is consistent. CPCC leads to a better coarse/coarser in-hierarchy performance, mostly better mid level performance, but not on fine level.

Dataset	Objective	FineAcc^1	MidAcc^1	CoarseAcc^0	CoarserAcc^0
MNIST^t	MTL	63.44 (11.39)	57.40 (4.35)	53.05 (1.69)	N/A
	MTLCPCC	**63.68 (4.89)**	58.56 (0.39)	56.94 (3.00)	
	Curr	56.12 (8.64)	50.69 (8.27)	**60.36 (3.38)**	
	CurrCPCC	57.87 (5.86)	58.09 (9.80)	59.91 (2.36)	
	SumLoss	62.10 (8.93)	**60.52 (6.53)**	52.44 (2.43)	
	SumLossCPCC	62.11 (14.76)	56.27 (6.53)	55.24 (2.18)	
LIVING17^t	MTL	30.92 (2.24)	35.00 (2.79)	55.39 (0.46)	70.75 (0.42)
	MTLCPCC	31.84 (3.04)	39.45 (3.56)	**61.76 (0.14)**	72.24 (0.58)
	Curr	29.70 (2.29)	32.71 (2.66)	54.33 (0.27)	70.20 (0.15)
	CurrCPCC	**32.36 (1.71)**	**40.15 (3.15)**	57.24 (0.77)	**72.84 (0.68)**
	SumLoss	31.26 (2.21)	32.09 (2.17)	54.41 (0.36)	70.46 (0.27)
	SumLossCPCC	32.07 (2.69)	38.91 (2.15)	57.18 (0.69)	72.79 (0.38)
ENTITY13^t	MTL	16.17 (1.87)	**22.18 (0.96)**	59.91 (0.53)	70.68 (0.29)
	MTLCPCC	13.04 (1.67)	18.74 (1.02)	61.76 (0.14)	72.47 (0.30)
	Curr	**16.29 (1.50)**	21.84 (1.15)	59.63 (0.49)	70.83 (0.34)
	CurrCPCC	12.71 (1.43)	18.96 (0.76)	**62.04 (0.31)**	**72.54 (0.28)**
	SumLoss	16.01 (1.64)	20.99 (0.34)	59.74 (0.56)	70.74 (0.42)
	SumLossCPCC	13.74 (1.57)	18.17 (0.43)	61.43 (0.59)	72.18 (0.53)
ENTITY30^t	MTL	21.84 (1.09)	22.98 (1.49)	47.17 (0.40)	60.73 (0.49)
	MTLCPCC	20.54 (0.66)	25.15 (1.33)	48.70 (0.43)	62.65 (0.28)
	Curr	**22.41 (1.01)**	22.84 (1.97)	47.37 (0.39)	60.83 (0.53)
	CurrCPCC	20.40 (0.95)	25.59 (1.79)	**49.70 (0.21)**	**63.40 (0.23)**
	SumLoss	21.51 (1.16)	22.35 (1.62)	46.51 (0.60)	60.44 (0.45)
	SumLossCPCC	21.48 (0.64)	**26.02** (0.93)	48.97 (0.42)	62.41 (0.39)
NONLIVING26^t	MTL	24.00 (1.64)	26.07 (0.59)	39.95 (0.64)	53.74 (0.53)
	MTLCPCC	23.83 (1.86)	27.35 (1.55)	41.24 (0.25)	54.95 (0.33)
	Curr	23.55 (1.44)	26.50 (1.37)	39.97 (0.46)	54.49 (0.60)
	CurrCPCC	23.93 (0.99)	**28.51 (2.51)**	**42.84 (0.60)**	**56.88(0.95)**
	SumLoss	**24.18 (1.40)**	25.75 (1.62)	40.02 (0.61)	53.78 (0.66)
	SumLossCPCC	24.05 (1.01)	27.37 (2.65)	41.92 (0.55)	55.64 (0.55)

```
CIFAR100
├── aquatic mammals + fish
│   ├── aquatic mammals
│   │   ├── beaver + dolphin
│   │   └── otter + seal + whale
│   └── fish
│       ├── aquarium fish + flatfish
│       └── ray + shark + trout
├── flowers + food containers
│   ├── flowers
│   │   ├── orchids + poppies
│   │   └── roses + sunflowers + tulips
│   └── food containers
│       ├── bottles + bowls
│       └── cans + cups + plates
├── fruit and vegetables + household electrical devices
│   ├── fruit and vegetables
│   │   ├── apples + mushrooms
│   │   └── oranges + pears + sweet peppers
│   └── household electrical devices
│       ├── clock + computer keyboard
│       └── lamp + telephone + television
├── household furniture + insects
│   ├── household furniture
│   │   ├── bed + chair
│   │   └── couch + table + wardrobe
│   └── insects
│       ├── bee + beetle
│       └── butterfly + caterpillar + cockroach
├── large carnivores + large man-made outdoor things
│   ├── large carnivores
│   │   ├── bear + leopard
│   │   └── lion + tiger + wolf
│   └── large man-made outdoor things
│       ├── bridge + castle
│       └── house + road + skyscraper
├── large natural outdoor scenes + large omnivores and herbivores
│   ├── large natural outdoor scenes
│   │   ├── cloud + forest
│   │   └── mountain + plain + sea
│   └── large omnivores and herbivores
│       ├── camel + cattle
│       └── chimpanzee + elephant + kangaroo
├── medium-sized mammals + non-insect invertebrates
│   ├── medium-sized mammals
│   │   ├── fox + porcupine
│   │   └── possum + raccoon + skunk
│   └── non-insect invertebrates
│       ├── crab + lobster
│       └── snail + spider + worm
├── people + reptiles
│   ├── people
│   │   ├── baby + boy
│   │   └── girl + man + woman
│   └── reptiles
│       ├── crocodile + dinosaur
│       └── lizard + snake + turtle
├── small mammals + trees
│   ├── small mammals
│   │   ├── hamster + mouse
│   │   └── rabbit + shrew + squirrel
│   └── trees
│       ├── maple + oak
│       └── palm + pine + willow
└── vehicles 1 + vehicles 2
    ├── vehicles 1
    │   ├── bicycle + bus
    │   └── motorcycle + pickup truck + train
    └── vehicles 2
        ├── lawn-mower + rocket
        └── streetcar + tank + tractor
```

(b) CIFAR hierarchy. The leaf nodes (100 fine classes) are evident from the mid level node names (they are simply separated by a +). We create the mid-level split based on alphabetical order. Label names are from `https://www.cs.toronto.edu/~kriz/cifar.html`.

```
LIVING17
├─ dummy6
│  └─ parrot
│     ├─ parrot
│     │  └─ macaw
│     └─ lory
│        └─ lorikeet
├─ dummy8
│  └─ salamander
│     ├─ salamander
│     │  └─ axolotl, mud puppy, Ambystoma mexicanum
│     └─ newt, triton
│        └─ eft
├─ serpentes
│  └─ snake, serpent, ophidian
│     └─ snake, serpent, ophidian
│        ├─ garter snake, grass snake
│        └─ night snake, Hypsiglena torquata
├─ chelonian, chelonian reptile
│  └─ turtle
│     ├─ turtle
│     │  └─ box turtle, box tortoise
│     └─ sea turtle, marine turtle
│        └─ leatherback turtle, leatherback, leathery turtle,
│           DFlatochelys coriacea
├─ saurian
│  └─ lizard
│     ├─ teiid lizard, teiid
│     │  └─ whiptail, whiptail lizard
│     └─ anguid lizard
│        └─ alligator lizard
├─ arachnid, arachnoid
│  └─ spider
│     └─ spider
│        ├─ black and gold garden spider, Argiope aurantia
│        └─ tarantula
├─ gallinaceous bird, gallinacean
│  └─ grouse
│     └─ grouse
│        ├─ ptarmigan
│        └─ prairie chicken, prairie grouse, prairie fowl
├─ crustacean
│  └─ crab
│     └─ crab
│        ├─ Dungeness crab, Cancer magister
│        └─ fiddler crab
└─ carnivore
   ├─ dog, domestic dog, Canis familiaris
   │  └─ dog, domestic dog, Canis familiaris
   │     ├─ Pekinese, Pekingese, Peke
   │     └─ bloodhound, sleuthhound
   ├─ wolf
   │  └─ wolf
   │     ├─ red wolf, maned wolf, Canis rufus, Canis niger
   │     └─ coyote, prairie wolf, brush wolf, Canis latrans
   ├─ fox
   │  └─ fox
   │     ├─ Arctic fox, white fox, Alopex lagopus
   │     └─ grey fox, gray fox, Urocyon cinereoargenteus
   ├─ domestic cat, house cat, Felis domesticus, Felis catus
   │  └─ domestic cat, house cat, Felis domesticus, Felis catus
   │     ├─ tiger cat
   │     └─ Egyptian cat
   └─ bear
      └─ bear
         ├─ American black bear, black bear, Ursus americanus,
         │  Euarctos americanus
         └─ sloth bear, Melursus ursinus, Ursus ursinus
```

(c) LIVING17 hierarchy (part 1).

```
LIVING17
├─ insect
│  ├─ beetle
│  │  └─ scarabaeid beetle, scarabaeid, scarabaean
│  │     ├─ dung beetle
│  │     └─ rhinoceros beetle
│  └─ butterfly
│     ├─ butterfly
│     │  └─ sulphur butterfly, sulfur butterfly
│     └─ nymphalid, nymphalid butterfly, brush-footed butterfly,
│        four-footed butterfly
│        └─ admiral
└─ primate
   ├─ monkey
   │  └─ monkey
   │     ├─ marmoset
   │     └─ titi, titi monkey
   └─ ape
      └─ ape
         ├─ orangutan, orang, orangutang, Pongo pygmaeus
         └─ gibbon, Hylobates lar
```

(d) LIVING17 hierarchy (part 2).

```
ENTITY13
├── structure, place
│   └── man-made structure, construction
│       ├── area
│       │   └── bell cote, bell cot
│       ├── dummy47
│       │   └── prison, prison house
│       ├── dummy67
│       │   └── fountain
│       ├── dummy46
│       │   └── planetarium
│       ├── barrier
│       │   └── breakwater, groin, groyne, mole, bulwark, seawall,
│       │       jetty
│       ├── reservoir
│       │   └── water tower
│       ├── signboard, sign
│       │   └── traffic light, traffic signal, stoplight
│       ├── dwelling, home, domicile, abode, habitation, dwelling
│       │   house
│       │   ├── cliff dwelling
│       │   ├── monastery
│       │   └── castle
└── paraphernalia
    ├── instrument
    │   ├── medical instrument
    │   │   └── syringe
    │   ├── percussion instrument, percussive instrument
    │   │   └── steel drum
    │   ├── piano, pianoforte, forte-piano
    │   │   └── upright, upright piano
    │   ├── bottle opener
    │   │   └── corkscrew, bottle screw
    │   ├── scientific instrument
    │   │   └── abacus
    │   ├── timer
    │   │   └── parking meter
    │   ├── tool
    │   │   └── lighter, light, igniter, ignitor
    │   ├── knife
    │   │   └── cleaver, meat cleaver, chopper
    │   ├── lock
    │   │   └── padlock
    │   ├── measuring instrument, measuring system, measuring device
    │   │   └── scale, weighing machine
    ├── equipment
    │   ├── ball
    │   │   ├── volleyball
    │   │   ├── basketball
    │   │   └── croquet ball
    │   ├── photographic equipment
    │   │   └── tripod
    │   ├── digital computer
    │   │   ├── hand-held computer, hand-held microcomputer
    │   │   └── notebook, notebook computer
    │   ├── electronic equipment
    │   │   ├── monitor
    │   │   └── projector
    │   ├── weight, free weight, exercising weight
    │   │   └── barbell
    │   ├── gymnastic apparatus, exerciser
    │   │   └── balance beam, beam
```

(e) ENTITY13 hierarchy (part 1).

```
ENTITY13
└─animal, animate being, beast, brute, creature, fauna
  ├─bird
  │ ├─passerine, passeriform bird
  │ │ └─indigo bunting, indigo finch, indigo bird, Passerina
  │ │   cyanea
  │ ├─hawk
  │ │ └─kite
  │ ├─phasianid
  │ │ └─quail
  │ ├─parrot
  │ │ └─African grey, African gray, Psittacus erithacus
  │ ├─cuckoo
  │ │ └─coucal
  │ ├─coraciiform bird
  │ │ └─bee eater
  │ ├─aquatic bird
  │ │ ├─spoonbill
  │ │ └─limpkin, Aramus pictus
  │ ├─coot
  │ │ └─American coot, marsh hen, mud hen, water hen, Fulica
  │ │   americana
  │ └─penguin
  │   └─king penguin, Aptenodytes patagonica
  └─reptile, reptilian
    ├─turtle
    │ └─mud turtle
    ├─sea turtle, marine turtle
    │ └─loggerhead, loggerhead turtle, Caretta caretta
    ├─agamid, agamid lizard
    │ └─agama
    ├─venomous lizard
    │ └─Gila monster, HelodFlata suspectum
    ├─chameleon, chamaeleon
    │ └─African chameleon, Chamaeleo chamaeleon
    ├─ceratopsian, horned dinosaur
    │ └─triceratops
    ├─snake, serpent, ophidian
    │ ├─thunder snake, worm snake, Carphophis amoenus
    │ ├─green snake, grass snake
    │ └─water snake
    └─cobra
      └─Indian cobra, Naja naja
    └─arthropod
      ├─insect
      │ ├─bee
      │ ├─walking stick, walkingstick, stick insect
      │ └─leafhopper
      ├─crab
      │ ├─rock crab, Cancer irroratus
      │ └─fiddler crab
      ├─beetle
      │ ├─tiger beetle
      │ └─ground beetle, carabid beetle
      └─spider
        ├─black and gold garden spider, Argiope aurantia
        ├─barn spider, Araneus cavaticus
        └─black widow, Latrodectus mactans
```

(f) ENTITY13 hierarchy (part 2).

```
ENTITY13
├── animal, animate being, beast, brute, creature, fauna
│   └── mammal, mammalian
│       ├── sea cow, sirenian mammal, sirenian
│       │   └── dugong, Dugong dugon
│       ├── carnivore
│       │   └── tiger, Panthera tigris
│       ├── dog, domestic dog, Canis familiaris
│       │   ├── Irish wolfhound
│       │   ├── Norwegian elkhound, elkhound
│       │   └── Samoyed, Samoyede
│       ├── domestic cat, house cat, Felis domesticus, Felis catus
│       │   ├── Persian cat
│       │   └── Siamese cat, Siamese
│       ├── ungulate, hoofed mammal
│       │   └── hippopotamus, hippo, river horse, Hippopotamus
│       │       amphibius
│       ├── goat, caprine animal
│       │   └── ibex, Capra ibex
│       └── monkey
│           └── colobus, colobus monkey
├── food, nutrient
│   └── produce, green goods, green groceries, garden truck
│       ├── vegetable, veggie, veg
│       │   ├── broccoli
│       │   ├── cauliflower
│       │   └── cucumber, cuke
│       ├── summer squash
│       │   └── spaghetti squash
│       ├── winter squash
│       │   ├── acorn squash
│       │   └── butternut squash
│       ├── sweet pepper
│       │   └── bell pepper
│       └── fruit
│           ├── fig
│           ├── corn
│           └── orange
└── apparel, toiletries
    └── garment
        ├── raincoat, waterproof
        │   └── trench coat
        ├── shirt
        │   └── jersey, T-shirt, tee shirt
        ├── skirt
        │   └── miniskirt, mini
        ├── swimsuit, swimwear, bathing suit, swimming costume,
        │   bathing costume
        │   ├── swimming trunks, bathing trunks
        │   └── bikini, two-piece
        ├── coat
        │   ├── abaya
        │   ├── cloak
        │   └── poncho
        ├── dress, frock
        │   └── gown
        └── garment
            └── military uniform
```

(g) ENTITY13 hierarchy (part 3).

```
ENTITY13
├── apparel, toiletries
│   └── accessory, accoutrement
│       ├── cap
│       │   └── bathing cap, swimming cap
│       ├── tights, leotards
│       │   └── maillot
│       ├── face mask
│       │   └── gasmask, respirator, gas helmet
│       ├── helmet
│       │   └── crash helmet
│       └── jewelry, jewellery
│           └── necklace
├── conveyance, transport
│   ├── wheeled vehicle
│   │   ├── cycles
│   │   │   └── unicycle, monocycle
│   │   ├── motorcycle, bike
│   │   │   └── motor scooter, scooter
│   │   ├── bus, autobus, coach, charabanc, double-decker, jitney,
│   │   │   motorbus, motorcoach, omnibus, passenger vehicle
│   │   │   ├── minibus
│   │   │   └── recreational vehicle, RV, R.V.
│   │   ├── car, auto, automobile, machine, motorcar
│   │   │   ├── racer, race car, racing car
│   │   │   └── jeep, landrover
│   │   ├── car, railcar, railway car, railroad car
│   │   │   └── passenger car, coach, carriage
│   │   ├── truck, motortruck
│   │   │   ├── trailer truck, tractor trailer, trucking rig, rig,
│   │   │   │   articulated lorry, semi
│   │   │   └── snowplow, snowplough
│   │   └── handcart, pushcart, cart, go-cart
│   │       └── shopping cart
│   └── craft
│       ├── airplane, aeroplane, plane
│       │   └── airliner
│       ├── passenger ship
│       │   └── liner, ocean liner
│       ├── boat
│       │   ├── fireboat
│       │   ├── yawl
│       │   └── speedboat
│       ├── sailboat, sailing boat
│       │   ├── catamaran
│       │   └── trimaran
│       ├── cargo ship, cargo vessel
│       │   └── container ship, containership, container vessel
│       ├── spacecraft, ballistic capsule, space vehicle
│       │   └── space shuttle
│       └── warship, war vessel, combat ship
│           └── aircraft carrier, carrier, flattop, attack aircraft
│               carrier
└── furnishing
    └── furniture, piece of furniture, article of furniture
        ├── baby bed, baby's bed
        │   └── bassinet
        ├── bed
        │   └── four-poster
        ├── cabinet
        │   └── file, file cabinet, filing cabinet
        ├── screen
        │   ├── shoji
        │   ├── fire screen, fireguard
        │   └── mosquito net
        ├── seat
        │   └── toilet seat
        ├── chair
        │   └── folding chair
        ├── chest of drawers, chest, bureau, dresser
        │   └── chiffonier, commode
        └── wall unit
            └── wardrobe, closet, press
```

(h) ENTITY13 hierarchy (part 4).

```
ENTITY30
├── garment
│   └── garment
│       ├── garment
│       │   └── apron
│       ├── coat
│       │   ├── academic gown, academic robe, judge's robe
│       │   └── fur coat
│       └── skirt
│           └── miniskirt, mini
├── bird
│   ├── passerine, passeriform bird
│   │   ├── passerine, passeriform bird
│   │   │   ├── brambling, Fringilla montifringilla
│   │   │   ├── goldfinch, Carduelis carduelis
│   │   │   └── water ouzel, dipper
│   │   └── titmouse, tit
│   │       └── chickadee
│   └── aquatic bird
│       ├── aquatic bird
│       │   ├── albatross, mollymawk
│       │   └── crane
│       ├── stork
│       │   └── white stork, Ciconia ciconia
│       └── sandpiper
│           └── red-backed sandpiper, dunlin, Erolia alpina
├── reptile, reptilian
│   ├── serpentes
│   │   ├── snake, serpent, ophidian
│   │   │   ├── thunder snake, worm snake, Carphophis amoenus
│   │   │   ├── king snake, kingsnake
│   │   │   └── garter snake, grass snake
│   │   └── black mamba, Dendroaspis augusticeps
│   │       └── green mamba
│   └── saurian
│       ├── iguanid, iguanid lizard
│       │   └── American chameleon, anole, Anolis carolinensis
│       ├── anguid lizard
│       │   └── alligator lizard
│       ├── venomous lizard
│       │   └── Gila monster, Heloderma suspectum
│       └── lacertid lizard, lacertid
│           └── green lizard, Lacerta viridis
└── arthropod
    ├── arachnid, arachnoid
    │   ├── arachnid, arachnoid
    │   │   ├── harvestman, daddy longlegs, Phalangium opilio
    │   │   └── scorpion
    │   └── spider
    │       ├── black widow, Latrodectus mactans
    │       └── barn spider, Araneus cavaticus
    ├── crustacean
    │   ├── crustacean
    │   │   ├── crayfish, crawfish, crawdad, crawdaddy
    │   │   └── hermit crab
    │   ├── crab
    │   │   └── Dungeness crab, Cancer magister
    │   └── lobster
    │       └── spiny lobster, langouste, rock lobster, crawfish,
                crayfish, sea crawfish
    └── insect
        ├── insect
        │   ├── fly
        │   ├── grasshopper, hopper
        │   └── lacewing, lacewing fly
        └── butterfly
            └── sulphur butterfly, sulfur butterfly
```

(i) ENTITY30 hierarchy (part 1).

```
ENTITY30
├── mammal, mammalian
│   ├── ungulate, hoofed mammal
│   │   ├── ungulate, hoofed mammal
│   │   │   ├── zebra
│   │   │   └── llama
│   │   ├── cattle, cows, kine, oxen, Bos taurus
│   │   │   └── ox
│   │   └── antelope
│   │       └── gazelle
│   ├── primate
│   │   ├── monkey
│   │   │   ├── baboon
│   │   │   └── howler monkey, howler
│   │   ├── lemur
│   │   │   └── Madagascar cat, ring-tailed lemur, Lemur catta
│   │   └── ape
│   │       └── chimpanzee, chimp, Pan troglodytes
│   └── carnivore
│       ├── carnivore
│       │   └── black-footed ferret, ferret, Mustela nigripes
│       └── dog, domestic dog, Canis familiaris
│           ├── Italian greyhound
│           ├── Bedlington terrier
│           └── basenji
├── fish
│   └── bony fish
│       ├── bony fish
│       │   └── tench, Tinca tinca
│       ├── salmon
│       │   └── coho, cohoe, coho salmon, blue jack, silver salmon,
│       │       Oncorhynchus kisutch
│       ├── butterfly fish
│       │   └── rock beauty, Holocanthus tricolor
│       └── scorpaenid, scorpaenid fish
│           └── lionfish
└── accessory, accoutrement, accouterment
    ├── headdress, headgear
    │   ├── clip
    │   │   └── hair slide
    │   ├── cap
    │   │   └── shower cap
    │   ├── hat, chapeau, lid
    │   │   └── bonnet, poke bonnet
    │   └── helmet
    │       └── pickelhaube
    ├── neckwear
    │   ├── brace
    │   │   └── neck brace
    │   ├── scarf
    │   │   └── feather boa, boa
    │   ├── necktie, tie
    │   │   └── Windsor tie
    │   └── neckwear
    │       └── bib
    └── footwear, legwear
        ├── footwear, legwear
        │   └── clog, geta, patten, sabot
        ├── tights, leotards
        │   └── maillot
        └── shoe
            ├── Loafer
            └── running shoe
```

(j) ENTITY30 hierarchy (part 2).

```
ENTITY30
├── appliance
│   └── home appliance, household appliance
│       ├── cooker
│       │   └── Crock Pot
│       └── home appliance, household appliance
│           ├── washer, automatic washer, washing machine
│           ├── microwave, microwave oven
│           └── vacuum, vacuum cleaner
├── craft
│   └── vessel, watercraft
│       ├── boat
│       │   └── lifeboat
│       ├── sailboat, sailing boat
│       │   └── trimaran
│       ├── warship, war vessel, combat ship
│       │   └── aircraft carrier, carrier, flattop, attack aircraft
│       │       carrier
│       └── cargo ship, cargo vessel
│           └── container ship, containership, container vessel
├── equipment
│   ├── electronic equipment
│   │   ├── telephone, phone, telephone set
│   │   │   └── pay-phone, pay-station
│   │   └── electronic equipment
│   │       ├── printer
│   │       ├── computer keyboard, keypad
│   │       └── microphone, mike
│   └── sports equipment
│       ├── sports equipment
│       │   ├── ski
│       │   └── racket, racquet
│       ├── ball
│       │   └── croquet ball
│       └── weight, free weight, exercising weight
│           └── dumbbell
└── instrument
    ├── kitchen utensil
    │   ├── kitchen utensil
    │   │   ├── spatula
    │   │   └── measuring cup
    │   ├── knife
    │   │   └── cleaver, meat cleaver, chopper
    │   └── pot
    │       └── coffeepot
    ├── measuring instrument, measuring system, measuring device
    │   ├── compass
    │   │   └── magnetic compass
    │   ├── timer
    │   │   └── parking meter
    │   ├── watch, ticker
    │   │   └── digital watch
    │   └── clock
    │       └── analog clock
    ├── musical instrument, instrument
    │   ├── percussion instrument, percussive instrument
    │   │   └── maraca
    │   ├── piano, pianoforte, forte-piano
    │   │   ├── upright, upright piano
    │   │   └── grand piano, grand
    │   └── wind instrument, wind
    │       └── French horn, horn
    └── tableware
        ├── bottle
        │   └── water bottle
        ├── bowl
        │   └── mixing bowl
        ├── glass, drinking glass
        │   └── beer glass
        └── jug
            └── water jug
```

(k) ENTITY30 hierarchy (part 3).

```
ENTITY30
├── instrument
│   └── tool
│       ├── lock
│       │   ├── combination lock
│       │   └── padlock
│       ├── tool
│       │   └── screw
│       └── pen
│           └── quill, quill pen
├── man-made structure, construction
│   ├── barrier
│   │   ├── gate
│   │   │   └── turnstile
│   │   ├── fence, fencing
│   │   │   └── picket fence, paling
│   │   └── barrier
│   │       ├── bannister, banister, balustrade, balusters, handrail
│   │       └── breakwater, groin, groyne, mole, bulwark, seawall,
│   │           jetty
│   └── building, edifice
│       ├── place of worship, house of prayer, house of God, house
│       │   of worship
│       │   └── mosque
│       ├── mercantile establishment, retail store, sales outlet,
│       │   outlet
│       │   ├── bookshop, bookstore, bookstall
│       │   └── butcher shop, meat market
│       └── dwelling, home, domicile, abode, habitation, dwelling
│           house
│           └── castle
├── wheeled vehicle
│   └── motor vehicle, automotive vehicle
│       ├── bus, autobus, coach, charabanc, double-decker, jitney,
│       │   motorbus, motorcoach, omnibus, passenger vehicle
│       │   └── school bus
│       ├── car, auto, automobile, machine, motorcar
│       │   ├── limousine, limo
│       │   └── convertible
│       └── minibike, motorbike
│           └── moped
├── cooked food, prepared food
│   └── dish
│       ├── dish
│       │   ├── pizza, pizza pie
│       │   └── potpie
│       ├── hamburger, beefburger, burger
│       │   └── cheeseburger
│       └── potato, white potato, Irish potato, murphy, spud, tater
│           └── mashed potato
└── produce, green goods, green groceries, garden truck
    ├── vegetable, veggie, veg
    │   ├── vegetable, veggie, veg
    │   │   ├── cucumber, cuke
    │   │   └── artichoke, globe artichoke
    │   ├── summer squash
    │   │   └── zucchini, courgette
    │   └── winter squash
    │       └── butternut squash
    └── fruit
        ├── fruit
        │   ├── strawberry
        │   ├── pineapple, ananas
        │   └── jackfruit, jak, jack
        └── eating apple, dessert apple
            └── Granny Smith
```

(l) ENTITY30 hierarchy (part 4).

```
NONLIVING26
├── armor
│   └── body armor, body armour, suit of armor, suit of armour,
│       coat of mail, cataphract
│       └── body armor, body armour, suit of armor, suit of armour,
│           coat of mail, cataphract
│           ├── breastplate, aegis, egis
│           └── bulletproof vest
├── bag
│   └── bag
│       └── bag
│           ├── plastic bag
│           └── purse
├── area
│   └── roof
│       └── roof
│           ├── dome
│           └── vault
├── barrier
│   └── fence, fencing
│       ├── fence, fencing
│       │   └── chainlink fence
│       └── rail fence
│           └── worm fence, snake fence, snake-rail fence, Virginia
│               fence
├── building, edifice
│   ├── dwelling, home, domicile, abode, habitation, dwelling house
│   │   └── dwelling, home, domicile, abode, habitation, dwelling
│   │       house
│   │       ├── monastery
│   │       └── palace
│   ├── mercantile establishment, retail store, sales outlet,
│   │   outlet
│   │   └── mercantile establishment, retail store, sales outlet,
│   │       outlet
│   │       ├── butcher shop, meat market
│   │       └── barbershop
│   └── outbuilding
│       ├── shed
│       │   └── apiary, bee house
│       └── outbuilding
│           └── greenhouse, nursery, glasshouse
├── electronic equipment
│   └── digital computer
│       └── digital computer
│           ├── desktop computer
│           └── laptop, laptop computer
├── garment
│   ├── coat
│   │   └── coat
│   │       ├── fur coat
│   │       └── lab coat, laboratory coat
│   └── skirt
│       └── skirt
│           ├── miniskirt, mini
│           └── hoopskirt, crinoline
├── headdress, headgear
│   └── hat, chapeau, lid
│       └── hat, chapeau, lid
│           ├── bearskin, busby, shako
│           └── bonnet, poke bonnet
├── kitchen utensil
│   └── pot
│       └── pot
│           ├── teapot
│           └── Dutch oven
└── measuring instrument, measuring system, measuring device
    └── timepiece, timekeeper, horologe
        ├── timer
        │   └── stopwatch, stop watch
        └── watch, ticker
            └── digital watch
```

(m) NONLIVING26 hierarchy (part 1).

```
NONLIVING26
├─ motor vehicle, automotive vehicle
│  ├─ bus, autobus, coach, charabanc, double-decker, jitney,
│  │  motorbus, motorcoach, omnibus, passenger vehicle
│  │  └─ bus, autobus, coach, charabanc, double-decker, jitney,
│  │     motorbus, motorcoach, omnibus, passenger vehicle
│  │        ├─ minibus
│  │        └─ trolleybus, trolley coach, trackless trolley
│  ├─ car, auto, automobile, machine, motorcar
│  │  └─ car, auto, automobile, machine, motorcar
│  │        ├─ Model T
│  │        └─ racer, race car, racing car
│  ├─ truck, motortruck
│  │  └─ truck, motortruck
│  │        ├─ pickup, pickup truck
│  │        └─ fire engine, fire truck
├─ musical instrument, instrument
│  ├─ percussion instrument, percussive instrument
│  │  └─ percussion instrument, percussive instrument
│  │        ├─ marimba, xylophone
│  │        └─ steel drum
│  ├─ wind instrument, wind
│  │  └─ wind instrument, wind
│  │        ├─ oboe, hautboy, hautbois
│  │        └─ sax, saxophone
│  ├─ keyboard instrument
│  │  ├─ piano, pianoforte, forte-piano
│  │  │  └─ grand piano, grand
│  │  └─ keyboard instrument
│  │     └─ organ, pipe organ
│  └─ stringed instrument
│     ├─ stringed instrument
│     │  └─ banjo
│     └─ guitar
│        └─ electric guitar
├─ seat
│  └─ chair
│     ├─ chair of state
│     │  └─ throne
│     └─ chair
│        └─ folding chair
├─ sports equipment
│  └─ ball
│     └─ ball
│        ├─ volleyball
│        └─ punching bag, punch bag, punching ball, punchball
├─ tableware
│  └─ bottle
│     └─ bottle
│        ├─ pop bottle, soda bottle
│        └─ beer bottle
├─ vessel, watercraft
│  ├─ boat
│  │  ├─ boat
│  │  │  └─ gondola
│  │  └─ sailboat, sailing boat
│  │     └─ trimaran
│  └─ ship
│     └─ ship
│        ├─ schooner
│        └─ pirate, pirate ship
└─ vegetable, veggie, veg
   └─ squash
      ├─ summer squash
      │  └─ spaghetti squash
      └─ winter squash
         └─ acorn squash
```

(n) NONLIVING26 hierarchy (part 2).