Abstract—Over the past decade, deep neural networks have proven to be adept in image classification tasks, often surpassing humans in terms of accuracy. However, standard neural networks often fail to understand the concept of hierarchical structures and dependencies among different classes for vision-related tasks. Humans on the other hand, seem to intuitively learn categories conceptually, progressively growing from understanding high-level concepts down to granular levels of categories. One of the issues arising from the inability of neural networks to encode such dependencies within its learned structure is that of subpopulation shift—where models are queried with novel unseen classes taken from a shifted population of the training set categories. Since the neural network treats each class as independent from all others, it struggles to categorize shifting populations that are dependent at higher levels of the hierarchy. In this work, we study the aforementioned problems through the lens of a novel conditional supervised training framework. We tackle subpopulation shift by a structured learning procedure that incorporates hierarchical information conditionally through labels. Furthermore, we introduce a notion of hierarchical distance to model the catastrophic effect of mispredictions. We show that learning in this structured hierarchical manner results in networks that are more robust against subpopulation shifts, with an improvement up to 3% in terms of accuracy and up to 11% in terms of hierarchical distance over standard models on subpopulation shift benchmarks.

Impact Statement—Subpopulation shift issues arise when a network trained on one set of sub-populations of labels (such as breeds of dogs) does not generalize to unseen sub-populations under the same label. This misclassification can grow catastrophic in critical decision-making scenarios such as autonomous driving. We propose to tackle this by learning richer representations built up hierarchically in a tree form, from top level concepts down to refining concepts, such as labeling ‘dog’ as a mammal followed by a carnivore, and lastly a dog. We introduce a conditional learning framework that allows hierarchical classification along with collaboration at each level. Representing labels as taxonomical hierarchies also allows us to measure the impact of mispredictions as the graphical distance between the true and predicted classes. We show that embedding hierarchical structure knowledge into the learned representation results in both better accuracy and lesser catastrophic misprediction rates on many subpopulation shift benchmarks.

Index Terms—Catastrophic mispredictions, hierarchical learning, representation learning, subpopulation shift.

I. INTRODUCTION

DEEP learning has been tremendously successful at image classification tasks, often outperforming humans when the training and testing distributions are the same. In this work, we focus on tackling the issues that arise when the testing distribution is shifted at a subpopulation level from the training distribution, a problem called subpopulation shift introduced recently in BREEDS [1]. Subpopulation shift is a specific kind of shift under the broader domain adaptation umbrella. In domain adaptation, the task of a classifier remains the same over the source and target domains, but there is a slight change in the distribution of the target domain [2], [3], [4], [5]. In the general setting, the target domain is a slightly changed version of the source domain. For example, an object detector that has been trained to detect objects during day time for a self-driving car application is used to perform the same task, but now on a shifted set of night time images. The task remains the same, i.e., to identify and detect objects, but the target domain (night time) is a shifted version of the source domain (day-time), provided all other conditions (such as weather, region, etc.) remain constant. There are other forms of shifts as well such as shifts in the marginal distribution of labels [6] or shifts under data imbalance [7]. These are broadly denoted as label and target shifts, respectively.

However, in the setting of subpopulation shift, both the source and target domains remain constant. The shift here occurs at a more granular level, that of subpopulations. Consider the source distribution described above, that of a self-driving car. Let us say the categories for classification included small-vehicles and large-vehicles. Under small-vehicles, the source set included samples of golf-car and race-car, and under large-vehicles, the source samples were from firetrucks and double-decker-buses. In the target domain for testing, the classes remain unaltered; the classifier is still learning to categorize vehicles into small or large categories. However, the testing samples are now drawn from different subpopulations of each class which were not present during training, such as coupe and sedan for small-vehicles and dumpster-truck and school-bus for large-vehicles.
Fig. 1. Example hierarchical representation of a custom subset of ImageNet. The classes for the classification task are at the intermediate level, denoted by “class.” The constituent subpopulations of each class are particular classes from the ImageNet dataset and are marked at the leaf level as “subpopulations.” The letter “S” denotes “Seen” distribution and “U” denotes “Unseen” shifted distributions. The colored arrows indicate the graphical distance from one leaf node to the other. This shows that mispredicting a Felidae as a Canis (two graph traversals) is less catastrophic than predicting the same as an Salamander (four graph traversals).

In addition, the current way of classification, in which each class is considered separate and independent of others, treats the impact of all mispredictions as equal. This is counter-intuitive, since a Husky and a Beagle are more similar to each other than to a Bull-frog. The impact of misclassifications becomes quite important in critical use cases. The cost of mispredicting an animate object for an inanimate object can be disastrous for a self-driving car. To address this, we introduce “catastrophic coefficient,” a quantitative measure of the impact of mispredictions that follows intuitively from a hierarchical graph. It is defined as the normalized length of the shortest path between the true and the predicted classes as per the graphical structure of the underlying hierarchy. We show that incorporating hierarchical information during training reduces the catastrophic coefficient of all considered datasets, under subpopulation shift.

We explicitly incorporate the hierarchical information into learning by reengineering the dataset to reflect the proposed hierarchical graph, a subset of which is sketched out in Fig. 1. We modify the neural network architectures by assigning intermediate heads (one fully connected layer) corresponding to each level of hierarchy, with one-hot labels assigned to the classes at each level individually, as shown in Fig. 2. We ensure that only samples correctly classified by a head are passed on for learning to the next heads (corresponding to descendants in the hierarchy graph) by a conditional learning mechanism. We first show results on a custom dataset we create out of ImageNet, and then scale up to three subpopulation benchmark datasets introduced by BREEDS [1] that cover both living and nonliving entities. We also show results on the BREEDS LIVING-17 dataset by keeping the hierarchical structure, but changing the target subpopulations to cover a more diverse range. We show that given a hierarchy, our learning methodology can result in both better accuracy and lower misprediction impact under subpopulation shift.

1) To the best of our knowledge, this is the first attempt to tackle the problem of subpopulation shift by hierarchical learning methods. Our method incorporates hierarchical information in two ways—allowing independent inference at each level of hierarchy and enabling collaboration between these levels by training them conditionally via filtering [8]. This ensures that each level only trains on samples that are correctly classified on all previous levels. This is similar to anytime inference ([9]), but the goal is no longer to enable efficient inference or early exit strategies, but to propagate conditional probabilities.

2) Framing the problem in a hierarchical setting allows us to quantify the misprediction impact, measured by the shortest hierarchical distance between the true and predicted labels for inference. While this has been considered in works on cost-sensitive classification [10], [11], we only use it as an evaluation metric to study the impact of hierarchies on subpopulation shift, instead of directly optimizing it.

3) We evaluate the performance of deep models under subpopulation shift and show that our training algorithm outperforms classical training in both accuracy and misprediction impact.

The rest of this article is organized as follows. In Section II, we describe relevant related work in subpopulation shift, hierarchical deep learning and domain adaptation and explain how our approach is different from them. In Section III, we describe in details our proposed approach to tackle the problem of subpopulation shift by learning hierarchical representation of datasets. In Section IV, we describe the experiments performed...
and present the results to show how hierarchical learning is beneficial for subpopulation shift. Finally, Section V concludes this article.

II. RELATED WORK

This section is broadly structured in four subsections. In the first subsection, we discuss the related work on subpopulation shift and the benchmark datasets associated with the problem. In the second subsection, we present a comprehensive study on hierarchical deep learning in the supervised learning setup and focus on the novelty of our approach in the formulation to tackle subpopulation shift. No previous hierarchical modeling technique has been used to study the problem of subpopulation shift problem. In the third subsection, we briefly discuss some notable works in domain adaptation which is a super set of the subpopulation shift problem. Lastly, in the fourth subsection, we present a discussion of different methods to learn hierarchical representations going beyond the supervised setting.

A. Subpopulation Shift

Subpopulation shift is a specific variant of domain adaptation where the models need to adapt to unseen data samples during testing, but the samples arrive from the same distribution of the classes, changed only at the subpopulation levels. BREEDS [1] introduced the problem of subpopulation shift along with tailored benchmarks constructed from the ImageNet [12] dataset. WILDS ([13]) provides a subpopulations shift benchmark but for toxicity classification across demographic identities. Cai et al. [14] tackled the problem through a label expansion algorithm similar to Li et al. [15] but tackled subpopulation shift by using the FixMatch [16] method. The algorithm uses semisupervised learning concepts such as pseudolabeling and consistency loss. Cai et al. [14] expanded upon this and showed how consistency-based loss is suitable for tackling the subpopulation shift problem. But these semisupervised approaches required access to the target set, albeit unlabelled, as the algorithm made use of these unlabelled target set to further improve upon a teacher classifier. Bukchin et al. [17] studied these subpopulation shift benchmarks for the problem of coarse-to-fine few shot learning (C2FS) task. As the name suggests, the method relies on training with coarse labels initially while adapting to finer labels with a few samples per label. This work introduced an angular normalization module known as ANCOR to effectively combine supervised interclass and semisupervised training of intraclass features. Liang et al. [18] introduced the incremental subpopulation shift problem, by adapting to unseen classes by learning on the seen classes. The problem focuses on learning new unseen classes while minimizing catastrophic forgetting of old seen classes. We restrict ourselves to the supervised training framework where we have no access to the target samples. Moreover, we tackle the subpopulation shift problem by incorporating hierarchical information into the models.

B. Hierarchical Deep Learning

Hierarchical modeling is a well-known supervised learning strategy to learn semantic concepts in vision datasets. Under this section, we cover methods that are shown on smaller datasets under small-scale methods and works that show results on ImageNet-scale datasets as large-scale methods.

Large-scale hierarchical methods: Hierarchical modeling is a well-known supervised learning strategy to learn semantic concepts in vision datasets. Under this section, we cover methods that are shown on smaller datasets under small-scale methods and works that show results on ImageNet-scale datasets as large-scale methods. Yan et al. [19] introduced hierarchical deep convolutional neural network (HD-CNN), which uses a base classifier to distinguish between coarser categories whereas for distinguishing between confusing classes, the task is pushed further downstream to the fine category classifiers. HD-CNN was novel in its approach to apply hierarchical training for large scale datasets but suffers from a different scalability problem. Its training requires copies of network parts for each subtree, and therefore, the network size continues to grow with bigger hierarchies. Furthermore, there is sequential pretraining, freezing, training, and fine-tuning required for each level of hierarchy, and hence, the authors limit their hierarchies to a depth of 2. Deng et al. [20] showed that the classification performance can be improved by leveraging semantic information as provided by the WordNet hierarchy. Deng et al. [21] further introduced hierarchy and exclusion graphs to capture semantic relations between two labels (parent and children). Although this work re-labels leaf nodes to intermediate parent nodes, they train models only on the leaf node labels (single label). Blocks [22] visually demonstrates via confusion matrices how learning hierarchies is an implicit method of learning for convolutional neural networks and similar classes are mapped close to one another along the diagonal of the learnt confusion matrix. Song et al. [23] shows how multiple heads of a neural network can collaborate among each other in order reach a consensus on image classification tasks. Verma et al. [24] introduced a dataset with hierarchical labels for human pose classification known as Yoga-82 and trains hierarchical variants of DenseNet [25] to benchmark classification accuracy on this set. In contrast, our work can be extended to multiple levels of hierarchy without the need for changing architecture, while employing a conditional training approach to link multiple labels of a single image as per the provided hierarchy. We show, by utilizing the hierarchy in this manner we are able to mitigate the effect of subpopulation shift, both under accuracy and impact of mispredictions. Hierarchical inference has often been used to enable efficient inference strategies in the scheme of anytime inference algorithms ([8], [9]). Additionally, [8] can also learn a label hierarchy. However, with these the aim is to make the inference pipeline more efficient by exiting early for easier examples. We perform inference at all levels of hierarchy to learn coarse-to-fine grained features to help with subpopulation shift.

Small-scale hierarchical methods: Branch convolutional neural networks (B-CNN) [26] learns multilevel concepts via a branch training strategy through weighted loss of the individual branches on small-scale datasets. HCNN [27] leverages hierarchical information to learn coarse to fine features on the Fashion-MNIST [28] dataset. Condition CNN [29] learns a conditional probability weight matrix to learn related features to help classification results on Kaggle fashion product.
images dataset. VT-CNN [30] introduces a new training strategy that pays attention to more confusing classes in CIFAR-10 and CIFAR-100 based on a confusion visual tree (CVT) that captures semantic level information of closely related categories. Inoue et al. [31] showed slight improvements on B-CNN by providing hierarchical semantic information to improve fine level accuracy on CIFAR-100 and Fashion-MNIST. Coarse-to-fine convolutional neural network (CF-CNN) [32] proposes a multilevel label augmentation method along with a fine and several coarse sub-networks to improve upon corresponding base networks. In our experiments, we provide an approach to hierarchically train deep models which scales to ImageNet based subpopulation shift benchmarks.

C. Domain Adaptation

Domain adaptation and its variants are a well studied set of problems in deep learning. One direction of works [4], [5], [33], [34], [35], [36], [37] is aimed at tackling the problem of adapting to target domains by learning on a selective set of samples from the target domain itself. Another line of work aims to match the source and target distributions in the feature space [38], [39], [40], [41]. The main motive behind these set of works is to tackle out-of-support domain adaptation tasks by sharing a common representation between the two. To adapt to newer environments, deep models are trained gradually to make them more suitable for transition to these newer environments [38], [42], [43], [44], [45]. Domain generalization enables the use of multiple different environments during training, but requires having a prior knowledge on the target distribution [46], [47], [48], [49].

We on the other hand, focus on a more specific problem of distribution shift, wherein the shift occurs at a subpopulation level in the target domain.

There are works done to learn similarity metrics in the context of hierarchical settings such as in [10]. A recent work, [11] introduced a similar notion of impact of mispredictions. [50] designed a theoretical framework for hierarchical image classification with a hierarchical cross-entropy model to show a slight improvement over [11]. On the other hand, we use the misprediction distance only as an evaluation metric to quantify the impact of our conditional training framework on the degree of catastrophic predictions, and instead look at the role that hierarchical learning plays in mitigating issues across domain shifts during inference.

D. Additional Related Work

Hierarchical semantic embedding: DeVise [51] presents a deep visual-semantic embedded model which learns similarity between classes in the semantic space both from images as well as unannotated text. Barz et al. [52], [53] demonstrated how prior knowledge can be leveraged based on hierarchy of classes such as WordNet to learn semantically discriminating features. Such learnt class embeddings projected on a unit hypersphere proved to be beneficial for both novel class predictions as well as image retrieval tasks.

Hierarchical learning for in nonsupervised approaches: Tree-CNN [54] tackles the incremental learning problem where the model expands as a tree to accommodate new classes. Zheng et al. [55] and Qu et al. [56] tackled the problem of metric learning via hierarchical concepts on large scale image datasets. Chen et al. [57] applied a semisupervised approach to learn cluster level concepts at higher level of a hierarchy and categorical features at leaf node levels. Jiang et al. [58] proposed a conditional class-aware Meta learning framework that conditionally learns better representations through modeling inter-class dependencies. Seo et al. [59] incorporated a hierarchical semantic loss function together with a confidence estimator to improve performance of zero-shot learning in terms of hit@k accuracy. Works such as McClelland et al. [60] and Saxe et al. [61] tried to understand the importance of hierarchical learning from a theoretical perspective and demonstrated an implementation on a neural network based model.

Hierarchical learning for interpretability: Interpreting predictions from CNNs has been key in understanding what features models look at in order to make predictions. Zhang et al. [62] provided a semantic as well as quantitative explanations for CNN predictions based on a decision tree in a coarse-to-fine manner at different fine-grained levels. Building on this concept, Hase et al. [63] introduced a model that leverages a predefined taxonomy to explain the predictions at each level of the taxonomy essentially showing how a Capuchin is gradually classified first as an animal, followed by a primate and finally as a Capuchin as per the hierarchy.

Applications of hierarchical learning: Dhall et al. [64], [65] showed how an image classifier augmented with hierarchical information based on entailment cone embeddings outperforms flat classifiers on an Entomological Dataset. Pham et al. [66] took advantage of the relationship between diseases in chest X-rays to learn conditional probabilities through image classifiers. Taoufiq et al. [67] adapted a similar approach to learn urban structural relationships.

III. METHODOLOGY: Hierarchies to Mitigate the Effect of Subpopulation Shift

In this section, we describe in details the proposed approach to tackle the problem of subpopulation shift. We begin by introducing some notations and formalizing the problem. We provide the perspective of using hierarchical learning as a method to mitigate the subpopulation shift problem. We then explain how vision datasets can be expressed as hierarchical trees with hot-encoded labels establishing relationships among the entities. We finally provide our proposed conditional training algorithm in Section III-F using modified deep neural network architectures outlined in Section III-E. We further introduce a notion of hierarchical distance in Section III-D which is used as an evaluation metric in the experiments performed, outlined in Section IV.

A. Subpopulation Shift

As described in Section I, subpopulation shift is a specific branch of the broader domain adaptation problem. In subpopulation shift, the training and the testing distributions differ at the level of subpopulations. Let us focus on an n-way classification problem, with each class denoted by i; i = {1, 2, ..., n}. The data consisting of image-label pairs for the source seen and the target
B. Hierarchical View to Tackle Subpopulation Shift

We tackle the subpopulation shift problem by explicitly incorporating hierarchical knowledge into learning via labels. Intuitively, if a neural network can grasp the concept of structural hierarchies, it will not overfit to the observed subpopulations. Instead, the network will have a notion of multiple coarse-to-fine level distributions that the subpopulation belongs to. The coarser distributions would likely cover a much larger set of distributions, hopefully helping in generalization under shift. For instance, a network trained with the knowledge that both a fire-truck and a race-car fall under vehicles, and a human and a dog fall under living things, will not overfit to the particular subpopulation but have a notion of vehicles and living things. This will allow it to generalize to a newer large-vehicle such as a school-bus and predict it as a vehicle rather than a living thing, since the network has learned a much broader distribution of vehicles one level of hierarchy above. Even if there is a misprediction, it is more likely to be at the lower levels of hierarchy, confusing things that are less catastrophic to mispredict.

C. Vision Datasets as Hierarchical Trees

ImageNet [12] is a large-scale image database collected on the basis of an underlying hierarchy called WordNet [68]. It consists of twelve different subtrees created by querying synsets from the WordNet hierarchy. To motivate the problem of subpopulation shift, we create two custom datasets from ImageNet, which are shifted versions of each other at the subpopulation level. The datasets have a balanced hierarchical structure of depth 3 as shown in Fig. 1, starting from coarse concepts such as mammals and amphibians at a higher level, to fine grained specific subpopulations at the leaf nodes.

Given a tree, we start at the root node and traverse downwards to the first level of hierarchy, which consists of superclasses such as mammals, fish, reptiles, etc. The custom dataset, for instance, has five superclasses, labeled 0−4. Next we traverse to the
Batch Conditional Training

Batch Size = 4

Conv Block 1  Conv Block 2  Conv Block 3  Conv Block 4

Conditional-Loss_1 = Loss_1
Conditional-Loss_2 = Loss_2 \times V_{1:2}
Conditional-Loss_3 = Loss_3 \times V_{2:3} = Loss_3 \times V_{2:3} \times V_{1:2}

Fig. 3. Practical implementation of conditional training for a batch of images. The validity mask serves to ensure that the blocks corresponding to a particular level are trained only on the instances that are correctly classified at the previous level. For example, validity masks $V_{1:2}$ represent the propagation of correctly classified instances from level $l_1$ to $l_2$, and contain a 1 where the instance was correctly classified by all levels between $l_1$ and $l_2$ and 0 otherwise. We implement conditional training via multiplication of losses with the corresponding validity masks.

level of classes. These are the actual tasks that the network has to classify. At this level, finer concepts are captured, conditioned on the previous level. For instance, the task now becomes: given an amphibian, is it a frog or a salamander; or given a bird, is it aquatic or aviatory. Each superclass in our custom dataset has only two classes, making up the $n = 10$ classes for classification. This level has one-hot encoding of all ten classes. Thus, the categorical labels are presented in a levelwise concatenated format as shown in Fig. 1. The label for frog is “4 → 8,” with the label 4 encoding that it belongs to the superclass of amphibians and 8 encoding that conditioned on being an amphibian, it is a frog. The models only see labels till the class level; the subpopulations labels are hidden from the networks. Finally, we reach the leaf nodes of the tree, where there are three subpopulations per class (figure only shows 1 from seen and unseen distributions). This overall encoding represents each label as a path arising from the root to the classes. The class labels always occur at level = depth − 1, one level above the subpopulations. For datasets such as LIVING-17 with a depth = 4, classes occur at level = 3 and we show an instance of this hierarchy in Fig. 2.

Accuracy and catastrophic coefficients are reported for the ten classes, similar to BREEDS. The custom trees are simple and balanced, capturing the hierarchical structure found in the dataset. We use them to lay the foundations on which we implement our conditional training framework. The two custom datasets are flipped versions of each other, created by keeping the hierarchical structure fixed. In one dataset, one subpopulation set becomes the seen distribution whereas the other one becomes the unseen one, and this is reversed for the second dataset. We then show how our method translates well to complicated hierarchies such as the LIVING-17, NonLIVING-26, and ENTITY-30 subpopulation shift benchmarks. This illustrates that our algorithm is compatible with any hierarchy chosen according to the task of interest.

D. Catastrophic Distance

In this section, we introduce the concept of catastrophic coefficient as a measure of the impact of misprediction. It is the shortest graphical distance between the true label and the predicted label in our hierarchy, normalized by the number of samples. It implicitly quantifies whether there is a notion of semantic structure in the model’s predictions. Subpopulation shift occurs at a lower level of a hierarchical tree where unseen subclasses are introduced during evaluation. So, if the hierarchically trained networks can grasp the concepts of superclasses and classes, the mispredictions during the shift will not be catastrophic. This is because they will tend to be correct at the higher levels, and hence ‘closer’ to the ground truth node in terms of graph traversal.

Neural networks trained via standard supervised learning have no explicit knowledge of inter-class dependencies. Thus, for flat models, mispredicting a specific subbreed of a dog as a subbreed of a cat is as catastrophic as mispredicting the same as a specific species of a snake. Graphical distance between the true and predicted classes intuitively captures the catastrophic impact of a misprediction and accounts for the semantic correctness of the prediction. This serves as an additional metric to accuracy for evaluating the performance of models under subpopulation shifts. In addition, it illustrates that the improvement in accuracy is truly due to incorporating better hierarchical information, rather than model architecture changes or the conditional training framework. It is pictorially illustrated by the colored arrows in Fig. 1.

A higher graphical distance between a misprediction and its ground truth signifies a more catastrophic impact. We average the graph traversal distances of all predictions ($\gamma = 0$ if sample classified correctly) over the entire dataset and call it the catastrophic coefficient, thus quantifying the impact of mispredictions for a network-dataset pair. Formally, let $g_k$ be the graph traversals needed for sample k in the shortest path between its...
true and predicted label. Let there be $N$ samples for evaluation. Then, the catastrophic coefficient is defined as $Cat = \frac{\sum_{k=1}^{N} y_k}{N}$. We note that we use this distance just to evaluate, and not during training. For evaluation, we take the final classifier level predictions and run it via our graph to check for distances, irrespective of whether they have been shown the hierarchy or not.

**E. Architecture**

We modify the standard ResNet [69] architectures to make them suitable for our conditional training framework. Since our network makes classification decisions at each level of the hierarchy, we introduce a separate head to predict the one-hot encoded vectors at each level. In a hierarchical subtree, the concept of a dog class is represented as a mammal at the superclass level, a carnivore at the class level and a dog at the subclass level. We want to train a multiheaded network where each head is trained on levelwise concepts starting from the superclass level, all the way down to the subclass level maintaining the path in the subtree. Thus, if we want to represent the hierarchical concept of a dog as shown in Fig. 2, the Head$_1$ of our network should predict if it is a mammal (superclass), Head$_2$ to predict if it is a carnivore (class) and finally Head$_3$ to predict that it is a dog (subclass). Convolutional neural networks learn coarse to fine features as they go deeper, capturing local concepts of images in the early layers, and global concepts in the later layers [70]. This lines up well with our hierarchical structure, and hence, we connect the different heads at different depths of the model. The concept is pictorially depicted in Fig. 2. Since we use residual networks in our work, the individual convolutional blocks here are residual blocks. The locations of these heads are determined experimentally. We got best results with Head$_1$ attached after the third residual block, Head$_2$ and Head$_3$ after the fourth residual blocks for the subtree shown in Fig. 2. We further want to ensure collaboration between these heads, done via a conditional training approach which we describe next.

**F. Conditional Training Details**

Here, we describe the conditional training framework, illustrated in Fig. 3, which is independent of subpopulation shift. Let us assume we have 3 levels in our hierarchy, with the levels enumerated by $l = 1, 2, 3$. Let the labels at each of these levels (treated as one-hot) be denoted by $y_l$. Let $F$ be the neural network that we pass a batch of images $X$ to. Here, $X \in \mathbb{R}^{B \times d}$ where $B$ is the batch-size and $d$ is the dimension of the input data. Further, let $F_l$ be the neural network up to head $l$, corresponding to predicting at level $l$ of our hierarchical graph. Note that all $F_l$ have overlap since the neural network is shared, rather than an ensemble, as illustrated in Fig. 2. The conditional loss at head $l$, $L_l$ is calculated as

$$L_l = CrossEntropy(F_l(x), y_l) \ast (V_{l-1})$$

where $V_{l-1}$ is the validity mask. $V_l \in \mathbb{R}^B$ contains all zeros, except ones at locations where the samples have been correctly classified by all heads until the head at level $l$. Each $V_l$ is generated by elementwise multiplication of the individual constituent masks, $[V_{l-2} \ast V_{l-3} \ast \ldots \ast V_{l-1}]$, allowing for an incorrect classification at any head to block further propagation of the incorrect instance to all deeper heads, shown pictorially in Fig. 3. This validity mask is what enforces our conditional training framework. Multiplying this validity mask with the current head’s loss before backpropagation ensures that learning for $F_l$ only occurs on samples that are meaningful at that head. In other words, $V$ propagates only correctly classified samples, as per its name. This ensures that the prediction at the $l-th$ head is not just $p(y_l)$, but $p(y_l | F_k(x) = y_k) \forall k = 1, 2, \ldots, l - 1$. In other words, it allows each head’s outcome to represent the probability of the current head’s prediction given that prediction of all levels until the current one were correct, allowing the network to learn progressively refining predictions. The network until a particular head is trained progressively on the conditional loss corresponding to that head. That means that during backpropagation, each layer get gradients from all conditional losses of heads located after that layer. This allows us to learn a shared backbone, but progressively refine features from coarser to finer as pertaining to the hierarchy.

**IV. EXPERIMENTS AND RESULTS**

In Section III, we described in detail our conditional training framework, where we train multiheaded networks to incorporate the notion of hierarchies in vision datasets. In this section, we empirically demonstrate how models trained with our approach perform better under subpopulation shift than models trained in a traditional flat learning setup. As a proof of concept, we first show results on two custom datasets created by querying the ImageNet on living entities. We then show the efficacy of our approach by expanding to three subpopulation shift benchmarks introduced in BREEDS (LIVING-17, NONLIVING-26, and ENTITY-30). Each of these subpopulation shift benchmarks have varying structures in terms of depth and width and each captures a diverse set of relationships among their entities. We explain each benchmark in details in the following, corresponding subsections. We compare our approach with a baseline model trained in the classical manner on all classes without any hierarchical information. We additionally compare with B-CNN [26], trained as per the branch training strategy outlined by the authors. In Section II, we mentioned HD-CNN [19] in terms of its novelty in training hierarchical deep models but as mentioned, it suffers from the issue of scalability and memory footprint with expanding hierarchies. The method is limited to hierarchies of depth $2$, whereas each subpopulation benchmark exhibits trees of depth $3$ or more. The method requires training one coarse classifier and multiple fine classifiers depending on how many coarse categories there are. With the current architectures of deep models, having multiple pretrained coarse and fine classifiers will vastly increase memory footprint and training time. Hence, we do not compare with HD-CNN.

**A. Overall Setup**

As mentioned, we consider subpopulation shift one level below the class level of a hierarchy. In this section we briefly describe the setup with the custom datasets as example. We
provide the exact details of each benchmark in the subsequent sections, summarized in Table I. Consider an n-way classification problem, with each class denoted by $i; \ i = \{1,2...n\}$. For our custom datasets, $n=10$. The total number of levels of hierarchy including the subpopulation levels, $l$, is 3. The $n$ classes are located at $l=2$ in our custom tree. Now, we create the shift by sampling subpopulations of each class $i$ from $s$ different subpopulations. For the custom datasets, $s=3$, and thus, for the custom datasets we have a 10-way classification problem with a total of $n \times s = 30$ subpopulations. More concretely, the subpopulations for class $i$ are distributed over $S_u^i$ (seen) and $S_s^i$ (unseen) domain. Let us consider $i = \{dogs\}$, $S_u^{dogs} = \{Bloodhound, Pekinese\}$ and $S_s^{dogs} = \{Great-Pyrenees, Papillon\}$. Thus, the learning problem is that by training on just the seen subpopulations $S_u^{dogs}$, the model should be able to identify that the unseen subpopulations of $S_s^{dogs}$, belong to $i = \{dogs\}$.

We use accuracy and catastrophic coefficient described in Section III-D to measure performance, both in the presence and absence of subpopulations shift. The higher the accuracy of a model, the better it is. On the contrary, the lower the catastrophic coefficient the better it is for a model. The number of graph traversals from the predicted node to the ground-truth node represents the value of a single misprediction impact, which varies from a minimum value of 0 (correct prediction) up to a maximum value of $2 \times (depth - 1)$ (worst case prediction, where predictions are made one level above subpopulations, and hence at a level of $(depth - 1)$). For example, for a dataset with $depth = 4$ such as LIVING-17 the worst case misprediction value for a single instance is 6, whereas for NonLIVING-26, the same is 8. Both accuracy and catastrophic coefficient are reported mainly under two different settings, differentiated by the subscript. The prefix of the subscript determines the domain the model was trained on and the suffix denotes the domain it is evaluated on. There are two combinations, “$s - s$” and “$s - u$,” with “s” representing seen data and “u” representing unseen data. “$s - u$” does not evaluate subpopulation shift, but shows the results of using our method as a general training methodology. It is trained on the standard training data of the seen domain, and evaluated on the validation set in the same seen domain. “$s - u$” evaluates results under subpopulation shift: training is performed on seen domain, and testing on unseen domain. Details of hierarchies and code will be made available soon.

### TABLE I

| Datasets     | Depth | Subpopulations ($s$) | Classes ($n$) |
|--------------|-------|----------------------|--------------|
| Custom       | 3     | 3                    | 10           |
| LIVING-17    | 4     | 2                    | 17           |
| Non-LIVING-26| 5     | 2                    | 26           |
| ENTITY-30    | 5     | 4                    | 30           |

details as mentioned in BREEDS [1], except bootstrapping. The classification task is on the $i$ classes, enumerated at the level of “classes” mentioned in the hierarchy. The subpopulation shift occurs one level below. The subpopulations labels are never shown to the network. The Hierarchical and B-CNN models, have been trained on the complete hierarchical information present in the tree, using our conditional training framework and the branch training strategy [26], respectively. The method oversees training to teach coarse to fine concepts as per the hierarchical structure of the target classes. The method invokes a weighted summation of a joint loss where each loss contribution comes from each branch (head) of a multiheaded network. There is a loss weight factor associated with each branch which dictates how much of each branch contributes toward the final loss as training continues. A head in our case is analogous to a branch in theirs. A branch represents a conceptual level in the hierarchy, and the training scheme dictates how much weight each branch contributes to the weighted loss as training goes on. In our conditional training framework, we sequentially train each head of our multiheaded network with the subsequent conditional loss as discussed in Fig. 2. In this manner, we teach the top–down hierarchical structure to networks and the conditional taxonomic relationships among its entities.

We use ResNet-18 as our network architecture backbones for modifications as mentioned in subsection III-E. For enumerating results, we use “Hierarchical-18” to denote a modified ResNet-18 model trained conditionally. Similarly, “Baseline-18” signifies a ResNet-18 architecture trained for flat classification on the $n$ categories found at the level of “classes” in the hierarchy. BCNN-18 refers to the modified ResNet-18 architecture trained via the branch training strategy [26].

### C. Results on Custom Datasets

In this section, we discuss the results on the two custom datasets, shown in Fig. 4 and Table II. We train the models on each dataset for 120 epochs with a batch size of 32, starting with a learning rate of 0.1, and a tenfold drop every 40 epochs thereafter. We do not use data augmentation on our custom datasets. All models have been trained on three random seeds each and the mean numbers are reported. $Acc_{s-u}$ and $Cat_{s-u}$ denote the accuracy and catastrophic coefficient of the model during the $s - u$ shift at the class level. As shown in Fig. 4 and Table II, our hierarchical-18 model performs better than the others, both in terms of accuracy and catastrophic coefficient, as well as both in the presence and absence of subpopulation shift. Moreover, the performance gap is significant under shift indicating that the imparted hierarchical information is helpful in correctly predicting unseen subpopulation classes. The models trained conditionally have ~4% – 5% improvement in terms of accuracy and an improvement of ~ (0.25 – 0.31) in hierarchical distance, translating into 10.0% and 7.9% improvement in terms of catastrophic coefficient over the flat baseline class level models under shift for the two custom sets. Fig. 4 and Table II highlight another interesting fact. Both the custom datasets have the same hierarchical structure and model the same semantic relationships among the entities; the difference is created by populating each set with different subpopulations. All models
MUKHERJEE et al.: ENCODING HIERARCHICAL INFORMATION IN NEURAL NETWORKS HELPS IN SUBPOPULATION SHIFT

Fig. 4. Results on Custom Datasets 1 and 2. Accuracy is shown on Table II below. The catastrophic coefficients are shown on the plot here. Our model outperforms the other in both accuracy and catastrophic coefficient on both “s-s” and “s-u” populations.

D. Results on LIVING-17

In this section, we discuss the results on the BREEDS LIVING-17 dataset, enumerated in Fig. 5 and Table III. For the LIVING-17 dataset, \( n = 17 \), depth is 4 and \( s = 2 \). The \( n \) classes are located at depth \( l = 4 \) and the subpopulations at \( l = 5 \), respectively. This subpopulation shift benchmark, introduced in BREEDS [1], captures finer details of hierarchy that encode richer relationships between the entities. We show that our methodology is applicable to complex hierarchies and outperforms class level baseline and BCNN-18 models both on \( \text{Acc}_{s-u} \) and \( \text{Cat}_{s-u} \). We train each architecture and model on five random seeds and report the mean numbers. We report numbers without bootstrapping, but follow all their other hyperparameters reported by BREEDS. Under the shift, hierarchical models achieve \( \sim 1.7\% \) – \( 3.5\% \) and \( 0.17 \) – \( 0.20 \) improvement in terms of accuracy and hierarchical distance, respectively, over other techniques. This results around 4% and 11% in terms of catastrophic coefficient over BCNN-18 and Baseline-18, respectively, as seen in Fig. 5.

Results on shifted LIVING-17: To cover a more diverse shift, we retain the hierarchy introduced in LIVING-17 but consider two more sets of different subpopulations. We call these LIVING-17-B and LIVING-17-C. These two shifted versions of the unseen set of LIVING-17 are formed by varying the \( S_i^w \) subclasses. We do this either by adding disjoint subclasses of the ImageNet [12] or by creating different combinations of the existing \( S_i^w \) with new disjoint subclasses. We reuse some of the \( S_i^w \) subclasses due to the unavailability of the same in the ImageNet database. All the subpopulations of \( i = \{ \text{wolf} \} \) from the ImageNet database have already been covered in the \( S_{\text{wolf}}^w \) and \( S_{\text{wolf}}^u \) set, so we just reuse the \( S_{\text{wolf}}^u \) in the sets B and C. \( \text{Acc}_{s-u}(B) \) denotes the model accuracy for the shift “s – u” from set A to B. As can be seen from Fig. 5 and Table III, Hierarchical-18 models have better accuracy and catastrophic coefficients than the other models for all three shifted sets. This shows that imparting hierarchical knowledge helps deep models to adapt to various degrees of the subpopulation shift.

E. Results on Nonliving-26

In this section, we describe results on the Nonliving-26 dataset, tabulated in Fig. 6 and Table IV. The dataset has \( n = 26 \) classes, a depth of 5 and number of subpopulation, \( s = 2 \). The \( n \) classes are located at depth \( l = 4 \) and the subpopulations at \( l = 5 \), respectively. For comparison, we train Baseline-18 and BCNN-18. We know that depth might hinder our conditional training process, since we limit samples that pass down from a level to the next contingent on their correct prediction at that level.
head. To test this, we create a collapsed version of this hierarchy. We collapse levels 1 and 2 into a single level and create a new hierarchy with the same amount of information and term this as L3 Hierarchical-18. All models are trained on three random seeds each and the mean numbers are reported.

As seen from Fig. 6 and Table IV, BCNN-18 outperforms our hierarchical model on “s-s” performance, while our framework performs better under both kinds of shift. In our conditional training framework, we only train subsequent heads if the previous heads have correctly classified the sample. As the depth of the hierarchical tree increases, fewer samples reach the final head for training, affecting the final classification performance on “s-s” models. Despite that, we outperform Baseline-18 and BCNN-18 both in terms of accuracy and catastrophic coefficient on the subpopulation shift “s-u” set. Since, the L3 Hierarchical-18 model is trained on one less level of hierarchical information, the final head gets to classify some more samples than Hierarchical-18 and has slightly better “s-s” performance.

F. Results on ENTITIY-30

We saw the effect of collapsing hierarchy with the previous set of experiments on NonLIVING-26. To realise the efficacy of our approach we perform similar experiments on the entire uncollapsed version as well as collapsed versions of the ENTITIY-30 dataset. The dataset has \( n = 30 \), a depth of 5 and \( s = 4 \). The \( n \) classes are located at depth \( l = 4 \) and the subpopulations at \( l = 5 \), respectively. The hierarchical tree encapsulates both living and nonliving entities and the more meaningful information is embedded between levels 3 and 4. Hence, we collapse the hierarchical information from levels 1 – 3 to a single level (2).

We train the L2 Hierarchical-18 and L2 BCNN-18 models on these two levels only whereas Hierarchical-18 and BCNN-18 models have been trained on the original uncollapsed (4) version of the hierarchy. The Baseline-18 models are trained flat on all the classes. All models have been trained on five random seeds each and the mean numbers are reported in Fig. 7 and Table V. The catastrophic coefficients are reported for “s-s” and “s-u” cases with the number of levels for evaluation in the hierarchy in brackets. We note that the L2 Hierarchical-18 has a comparable performance with Baseline-18 on “s-s” set but on the shifted unseen distribution, there is a boost in both accuracy and catastrophic co-efficient for all hierarchy aware models over

---

**TABLE IV**

| Model          | \( Acc_{s-s} \) | \( Acc_{s-u} \) |
|----------------|-----------------|-----------------|
| Baseline-18    | 88.39           | 42.13           |
| BCNN-18        | 88.1            | 42.4            |
| L3 Hierarchical-18 | 87.71          | 42.44           |
| Hierarchical-18 | 87.41           | 42.94           |

Corresponding catastrophic coefficients are shown in the plot above in Fig. 6. The L3 Hierarchy is the same hierarchy with the first two levels collapsed into a single level. The bold entities signify the best performing models in each case in terms of accuracy.

**TABLE V**

| Model          | \( Acc_{s-s} \) | \( Acc_{s-u} \) |
|----------------|-----------------|-----------------|
| Baseline-18    | 87.98           | 49.52           |
| L2 BCNN-18     | 87.75           | 49.57           |
| L2 Hierarchical-18 | 87.93         | 50.24           |
| BCNN-18        | 87.66           | 49.69           |
| Hierarchical-18 | 87.48           | 49.88           |

Corresponding catastrophic coefficients are shown on the plot above in Fig. 7. The bold entities signify the best performing models in each case in terms of accuracy.
the baseline one. This shows the importance of hierarchical training even for larger datasets with complicated hierarchy structure. We will also like to point to the fact that Hierarchical-18 models outperform the corresponding BCNN-18 models demonstrating the effect of the conditional training to tackle the subpopulation shift problem. Under both the collapsed (2) and uncollapsed (4) hierarchies, our models have less catastrophic mispredictions under both “s-s” and “s-u” distributions.

G. Discussions

In this subsection, we provide a brief summary of the key findings from the experiments performed. In terms of both accuracy and catastrophic coefficient, we show that the proposed hierarchical models are superior to the baseline class models and B-CNN models in tackling the subpopulation shift problem in all the five cases considered. We note that in three out of five sets of results, the trend of improvement in the subpopulation shifted (unseen) dataset was tracked by the unshifted (seen) dataset as well. However, the trend is not unanimous. For instance, improvements in accuracies track each other roughly in both custom datasets and LIVING-17 datasets, but not for nonLIVING-26 and ENTITY-30 datasets. We believe that the kind of shifted subpopulation itself has an impact on this, as evidenced by the different results we get by shifting one source to three targets in LIVING17-A, B, and C. We also believe that the classes where the shift occurs determine how easy the categorization under shift is, and hence see different trends for say, NonLIVING-26 and ENTITY-30 datasets. We have also seen for the latter two datasets how collapsing hierarchies in a meaningful manner can impact performances for the hierarchy aware methods both with the the proposed Hierarchical-18 models outperforming the BCNN-18 models for such larger subpopulation shift benchmarks as well.

V. Conclusion

In this article, we target the problem of subpopulation shift, which is a specific kind of shift under the broader umbrella of domain adaptation. The subpopulations that make up the categories for the classification task change between training and testing. For instance, the testing distribution may contain new breeds of dogs not seen during training, but all samples will be labeled “dog.” We note an implicit notion of hierarchy in the framing of the problem itself; in the knowledge of all constituent subpopulations sharing the common immediate ancestry. In line with this, we extend the notion of hierarchy and make it explicit to better tackle the issue of subpopulation shift. We consider the underlying hierarchical structure of vision datasets, in the form of both our own custom subsets and benchmark datasets for subpopulation shift. We incorporate this information explicitly into training via labeling each level with an individual one-hot label, and then encourage collaboration between multiple heads of a model via a conditional training framework. In this framework, each head is only trained on samples that were correctly classified at all levels before the present one. We further introduce a metric to capture the notion of semantic correctness of predictions. It uses the shortest hierarchical distance between the misprediction and the true label as per the hierarchy to quantify the catastrophic impact of mispredictions. We show that our hierarchy-aware conditional training setup outperforms flat baselines by around $1\%$ in terms of accuracy and $\sim (3-11)\%$ in terms of catastrophic coefficient over standard models across two custom datasets and three subpopulation shift benchmarks. We would like to mention one limitation to our approach. Our methodology requires access to hierarchical labels since this is a fully supervised learning framework. But we believe, this approach can be extended to the semisupervised or self-supervised setting as well, where one can leverage both the seen and unseen distributions and learn hierarchical representation of the unseen domain using a teacher signal from the seen distribution. Thus, the problem can be formalized as a domain adaptation approach similar to the work mentioned in Cai et al. [14]. Contrary to this work, our method can be used to learn representations across all levels of the given hierarchy and hence improve classification performance on the unseen distribution.

REFERENCES

[1] S. Santurkar, D. Tsipras, and A. Madry, “BREEDS: Benchmarks for subpopulation shift,” in Proc. Int. Conf. Learn. Representations, 2021.
[2] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, Deep Learning, vol. 1. Cambridge, MA, USA: MIT Press, 2016.
[3] J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, Dataset Shift in Machine Learning. Cambridge, MA, USA: MIT Press, 2009.
[4] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, “Adapting visual category models to new domains,” in Proc. 11th Eur. Conf. Comput. Vis., 2010, pp. 213–226.
[5] Y. Ganin and V. Lempitsky, “Unsupervised domain adaptation by back-propagation,” in Proc. 32nd Int. Conf. Mach. Learn., 2015, vol. 37, pp. 1180–1189.
[6] R. T. D. Combes, H. Zhao, Y-X. Wang, and G. J. Gordon, “Domain adaptation with conditional distribution matching and generalized label shift,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 19276–19289.
[7] Y. Li, M. Murias, S. Major, G. Dawson, and D. E. Carlson, “On target shift in adversarial domain adaptation,” in Proc. 22nd Int. Conf. Artif. Intell. Statist., 2019, pp. 616–625.
[8] J. Deng, S. Satheesh, A. Berg, and F. Li, “Fast and balanced: Efficient label tree learning for large scale object recognition,” in Proc. Adv. Neural Inf. Process. Syst., 2011.
[9] S. Karayev, M. Fritz, and T. Darrell, “Anytime recognition of objects and scenes,” in Proc. IEEE Comput. Vis. Pattern Recognit., 2014, pp. 572–579.
[10] N. Verma, D. Mahajan, S. Sellamanickam, and V. Nair, “Learning hierarchical similarity metrics,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2012, pp. 2280–2287.
[11] L. Bertinetto, R. Mueller, K. Tertikas, S. Samangoei, and N. A. Lord, “Making better mistakes: Leveraging class hierarchies with deep networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 12506–12515.
[12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in Proc. IEEE Comput. Vis. Pattern Recognit., 2009, pp. 248–255.
[13] P. W. Koh et al., “Wilds: A benchmark of in-the-wild distribution shifts,” in Proc. Int. Conf. Mach. Learn., 2021, pp. 5637–5664.
[14] T. Cai, R. Gao, J. Lee, and Q. Lei, “A theory of label propagation for subpopulation shift,” in Proc. Int. Conf. Mach. Learn., 2021, pp. 1170–1182.
[15] B. Li et al., “Rethinking distributional matching based domain adaptation,” 2020, arXiv:2006.13352.
[16] K. Sohn et al., “Fixmatch: Simplifying semi-supervised learning with consistency and confidence,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 596–608.
[17] G. Bukchin et al., “Fine-grained angular contrastive learning with coarse labels,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 8730–8740.
[45] Y. Chen, C. Wei, A. Kumar, and T. Ma, “Self-training avoids using spurious features under domain shift,” in Adv. Neural Inf. Process. Syst., vol. 33, pp. 21061–21071, 2020.

[46] M. Glifary, W. Kleijn, M. Zhang, and D. Balduzzi, “Domain generalization for object recognition with multi-task autoencoders,” in Proc. IEEE Int. Conf. Comput. Vis., 2015, pp. 2551–2559.

[47] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales, “Learning to generalize: Meta-learning for domain generalization,” in Proc. AAAI Conf. Artif. Intell., 2017.

[48] M. Arjoysky, L. Bottou, I. Gulrajani, and D. Lopez-Paz, “Invariant risk minimization,” 2019, arXiv:1907.02893.

[49] H.-T. Ye, C. Xie, T. Cai, R. Li, Z. Li, and L. Wang, “Towards a theoretical framework of out-of-distribution generalization,” in Proc. Adv. Neural Inf. Process. Syst., 2021.

[50] S. Shkodrani, Y. Wang, M. Manfredi, and N. Baka, “Unified we learn better: Harvesting learning improvements from class hierarchies across tasks,” 2021, arXiv:2107.13627.

[51] A. Frome et al., “Deive: A deep visual-semantic embedding model,” in Proc. Adv. Neural Inf. Process. Syst., 2013.

[52] B. Barz and J. Denzler, “Hierarchy-based image embeddings for semantic image retrieval,” in Proc. IEEE Winter Conf. Appl. Comput. Vis., 2019, pp. 638–647.

[53] B. Barz and J. Denzler, “Content-based image retrieval and the semantic gap in the deep learning era,” in Proc. Int. Conf. Pattern Recognit. Workshops, 2020, pp. 245–260.

[54] D. Roy, P. Panda, and K. Roy, “Tree-CNN: A hierarchical deep convolutional neural network for incremental learning,” Neural Netw., vol. 121, pp. 148–160, 2020.

[55] Y. Zheng, J. Fan, J. Zhang, and X. Gao, “Hierarchical learning of multi-task sparse metrics for large-scale image classification,” Pattern Recognit., vol. 67, pp. 97–109, 2017.

[56] Y. Qu et al., “Joint hierarchical category structure learning and large-scale image classification,” IEEE Trans. Image Process., vol. 26, no. 9, pp. 4331–4346, Sep. 2017.

[57] T. Chen, S. Lu, and J. Fan, “SS-HCNN: Semi-supervised hierarchical convolutional neural network for image classification,” IEEE Trans. Image Process., vol. 28, no. 5, pp. 2389–2398, May 2019.

[58] X. Jiang, M. Havaei, F. Varno, G. Chartrand, N. Chapados, and S. Matwin, “Learning to learn with conditional class dependencies,” in Proc. Int. Conf. Learn. Representations, 2019.

[59] S. J. Ieo and S. Kim, “Hierarchical semantic loss and confidence estimator for visual-semantic embedding-based zero-shot learning,” Appl. Sci., vol. 9, no. 15, 2019, Art. no. 3133.

[60] J. L. McClelland, S. Sadeghi, and A. M. Saxe, “A critique of pure hierarchy: Uncovering cross-cutting structure in a natural dataset,” 2016.

[61] A. M. Saxe, L. J. McClelland, and S. Ganguli, “Learning hierarchical categories in deep neural networks,” Cogn. Sci., 2013.

[62] Q. Zhang, Y. Yang, Y. N. Wu, and S.-C. Zhu, “Interpreting CNNs via decision trees,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 6261–6270.

[63] P. Hase, C. Chen, O. Li, and C. Rudin, “Interpretable image recognition with hierarchical prototypes,” in Proc. AAAI Conf. Hum. Comput. Crowdsourcing, 2019.

[64] A. Dhall, A. Makarova, O.-E. Ganea, D. Pavlov, M. Greff, and A. Krause, “Hierarchical image classification using entailment cone embeddings,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops, 2020, pp. 836–837.

[65] A. Dhall, “Learning representations for images with hierarchical labels,” 2020, arXiv:2004.00909.

[66] H. Pham, T. T. Le, D. T. Ngo, D. Q. Tran, and H. Q. Nguyen, “Interpreting chest x-rays via CNNs that exploit hierarchical disease dependencies and uncertainty labels,” 2021.

[67] S. Taoufik, B. Nagy, and C. Benedek, “Hierarchynet: Hierarchical CNN-based urban building classification,” Remote Sens., vol. 12, no. 22, 2020, Art. no. 3794.

[68] G. A. Miller, “WordNet: A lexical database for English,” Commun. ACM, vol. 38, no. 11, pp. 39–41, 1995.

[69] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2016, pp. 770–777.

[70] M. D. Matthew and R. Fergus, “Visualizing and understanding convolutional networks,” in Proc. Eur. Conf. Comput. Vis., 2014, pp. 818–833.