

What is a Tabby? Interpretable Model Decisions by Learning Attribute-Based Classification Criteria

Haomiao Liu¹, Member, IEEE, Ruiping Wang², Member, IEEE,
Shiguang Shan³, Senior Member, IEEE, and Xilin Chen⁴, Fellow, IEEE

Abstract—State-of-the-art classification models are usually considered as black boxes since their decision processes are implicit to humans. On the contrary, human experts classify objects according to a set of explicit hierarchical criteria. For example, “*tabby* is a *domestic cat* with stripes, dots, or lines”, where *tabby* is defined by combining its superordinate category (*domestic cat*) and some certain attributes (e.g., has stripes). Inspired by this mechanism, we propose an interpretable Hierarchical Criteria Network (HCN) by additionally learning such criteria. To achieve this goal, images and semantic entities (e.g., taxonomies and attributes) are embedded into a common space, where each category can be represented by the linear combination of its superordinate category and a set of learned discriminative attributes. Specifically, a two-stream convolutional neural network (CNN) is elaborately devised, which embeds images and taxonomies with the two streams respectively. The model is trained by minimizing the prediction error of hierarchy labels on both streams. Extensive experiments on two widely studied datasets (CIFAR-100 and ILSVRC) demonstrate that HCN can learn meaningful attributes as well as reasonable and interpretable classification criteria. Therefore, the proposed method enables further human feedback for model correction as an additional benefit.

Index Terms—Interpretable model, visual attributes, convolutional neural network, classification criteria

1 INTRODUCTION

WHILE state-of-the-art classification / categorization models [1], [2], [3], [4], [5], [6], [7], [8], [9], [10] achieve excellent accuracies on given object sets, most of them are purely trained to deal with a closed world. As a result, the classification processes of such models are implicit to humans, and incorporating prior knowledge (either from known objects or from human beings) into the model still remains an open research problem. Moreover, a major drawback of previous models is that they are easy to be cheated, for example as DeepFool [11]. As a result, it's hard to be trusted in vital tasks such as self-driving cars and medical care.

On the contrary, humans can explain the rationale behind their decisions on complex object recognition in an interpretable way, even though the decision process happens unconsciously. For example, for some classification tasks, especially those simple and familiar ones such as recognizing faces, humans can easily make decisions without explicit criteria. However, as shown in [12], the “criteria” (i.e., different characteristics of faces and the way they compose a whole face) have already been encoded into the human brain (neurons), and thus humans recognize faces

“without thinking” thanks to such implicit criteria. For more general perception tasks, take the object recognition task as an example, the most commonly adopted criteria system defines each category as the combination of its superordinate category and a set of visual attributes (as shown in Fig. 1, which is adopted by Wikipedia and many other encyclopedias). For example, a *tabby cat* is “a *domestic cat* that has a coat featuring distinctive stripes, dots, lines or swirling patterns” and a *domestic cat* is “a small, typically furry, carnivorous *mammal*, and the only domesticated species in the family *Felidae* (i.e., *feline*)” (Fig. 1b). By substituting the definition of *domestic cat* into the definition of *tabby cat*, *tabby cat* can also be defined as “a domesticated, small, furry, carnivorous *mammal* with stripes, dots, etc.” or equivalently “a domesticated, small *feline* with stripes, dots, etc.” Furthermore, by recursively tracing back to the root of the hierarchy as above, each category can be represented as a certain combination of just attributes, which is termed as the prototype of the corresponding category [13]. The above example shows that attributes can well characterize the nature of each individual category and the connections among categories. More importantly, attributes are not only human-understandable but also machine-learnable, making it possible to learn interpretable classification models that are in the same form as the expert-defined criteria. Besides, similarly to “category”, “attribute” is also a kind of semantic feature for clustering / classifying.

Recently, researchers have paid more and more attention to improve the interpretability of classification models [14], [15], [16], [17], [18], [19]. Although these methods have certainly achieved some successes, most of the interpretations obtained by these methods are not as explicit as the expert-

- H. Liu is with the Huawei EI Innovation Lab, Beijing 100085, China. E-mail: haomiao.liu@vipl.ict.ac.cn.
- R. Wang, S. Shan, and X. Chen are with the Key Laboratory of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, Beijing 100190, China. E-mail: {wangruiping, sgshan, xlchen}@ict.ac.cn.

Manuscript received 13 Nov. 2018; revised 10 Aug. 2019; accepted 3 Nov. 2019. Date of publication 20 Nov. 2019; date of current version 1 Apr. 2021. (Corresponding author: Xilin Chen.)

Recommended for acceptance by A. Gupta.

Digital Object Identifier no. 10.1109/TPAMI.2019.2954501

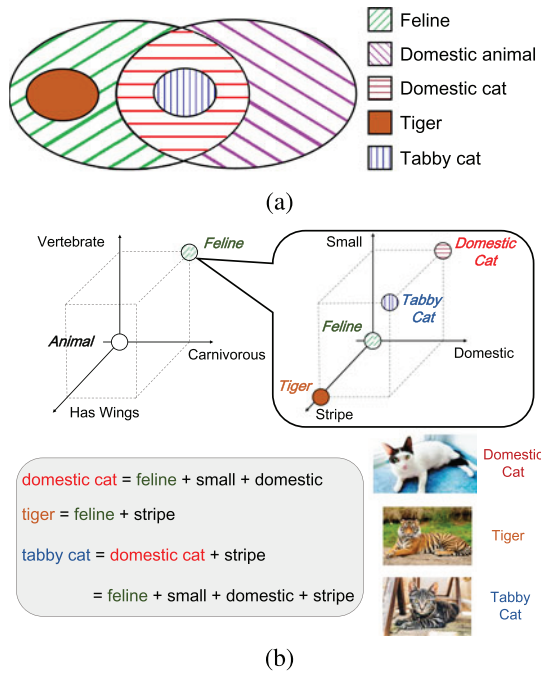


Fig. 1. An illustration of the attribute-based classification criteria system. (a) A simplified category hierarchy. (b) Example criteria on these hierarchically organized categories (i.e., *feline*, *domestic cat*, *tiger*, and *tabby cat*), which are defined by attributes like domestic, small, and stripe.

defined criteria described above, which limits their interpretability to some extent. One exception is the method proposed in [14]. [14] decomposes each category into the linear combination of its superordinate category and a series of manually defined attributes. Although this provides a method to embed predefined attributes, as the number of categories grows, some new attributes have to be developed for dealing with the new categories, which inevitably limits its scalability.

To tackle the scalability problem of manually defined attributes, we propose to automatically learn the attributes with only image-level category labels and class hierarchy, instead of relying on manually-defined ones. This is invaluable for open-world recognition. To this end, the method is

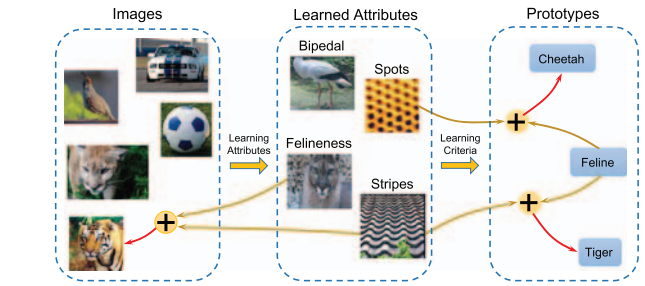


Fig. 2. The motivation of the proposed method. Visual attributes learned from images are used to represent the images, and such attributes are also used to characterize the connections between category prototypes, where the prototype of each category can be obtained by combining the prototype of its superordinate category and a certain set of attributes. By this means, the hierarchical classification criteria can be easily derived from the learned prototypes.

proposed to find the visual attributes that relate to each known category and how the category is described by these attributes as shown in Fig. 2. Specifically, the visual attributes are learned from images, and the images are represented by the learned attributes in return. Moreover, the learned attributes are also used to bridge the prototypes of the hierarchically organized categories, so that we could obtain the classification criteria that are similar to the expert-defined ones (e.g., *tiger* is a *feline* with stripes). Note that since the attributes are purely mined from images in the proposed method, we do not expect them to have an exact one-to-one correspondence with manually defined ones. For example, the “felineness” attribute illustrated in Fig. 2 has hardly been defined by humans but could be vital for defining the category *feline*.

Inspired from recognition/categorization in psychology and the successes of CNN models in object recognition and attribute-related tasks [16], [18], [20], [21], we propose a two-stream network named Hierarchical Criteria Network (HCN) as shown in Fig. 3. Specifically, the two streams embed images and category prototypes into a common space, where the bases of the space are visual attributes learned on the images. More concretely, the upper stream of the network takes images as inputs and extracts the image

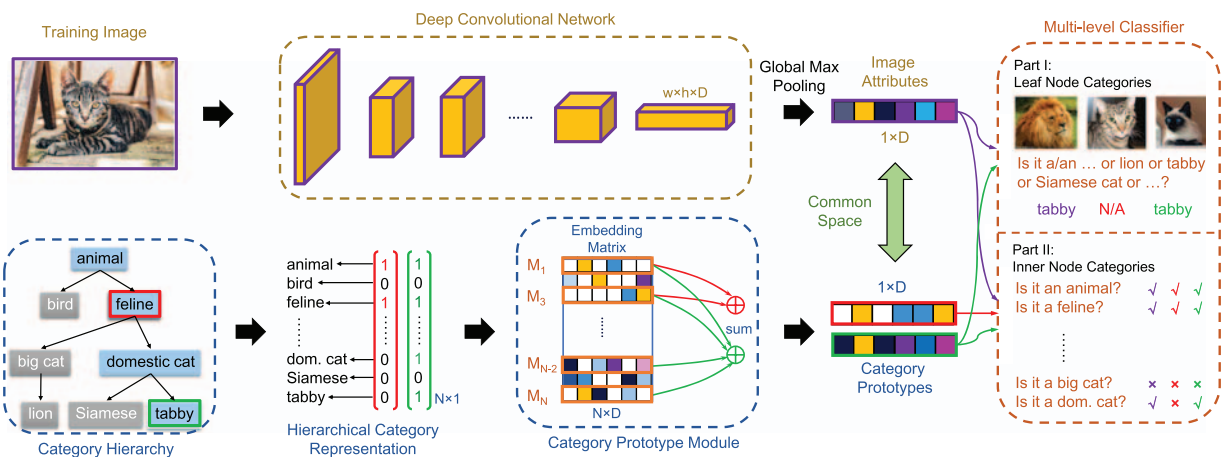


Fig. 3. The proposed HCN. To learn the hierarchical criteria between categories in terms of attributes, we propose a two-stream CNN structure, where the upper stream is designed to extract image attributes, while the lower stream learns prototypes for each category in the hierarchy. The model is trained to predict the corresponding hierarchical category labels using both the image samples and the category prototypes. For ease of optimization, the category prototypes are decomposed to the summation of rows in a hierarchical embedding matrix (each column represents an attribute) selected by the binary hierarchical category representations.

features as outputs, where each dimension of the feature is seen as an attribute. On the other hand, the lower stream is designed to learn prototypes for each category. For this purpose, the prototypes are embedded into the same space as image features so that the prototypes could also be represented by the learned attributes. Intuitively, each category is a special case of its superordinate category, and thus it has all attributes of its superordinate category. Based on this intuition, a set of consistency constraints are imposed on the prototypes to ensure that each category inherits all attributes from its superordinate categories, and also has some additional specialties. To train the model, both the image features and the category prototypes are fed into a shared classifier to accurately predict their corresponding hierarchical labels. In such a framework, by inspecting the difference between the prototypes of a category and its superordinate category, we could tell which attributes are the most significant differences between the two categories, and thus the classification criteria could be easily derived. More importantly, such criteria could not only be used to interpret the model predictions but also enable further human feedback for model correction. Experiments are conducted on two widely studied datasets (CIFAR-100 and ILSVRC), and the results validate that HCN improves the model interpretability and learns reasonable classification criteria.

2 RELATED WORKS

In HCN, CNN models are trained to learn attributes for hierarchical classification and interpreting the decision criteria. Therefore, HCN is mainly related to attribute learning, hierarchy-regularized classification, and network interpretation.

Attribute Learning. As a kind of human-understandable semantic descriptors [22], visual attributes are also machine-learnable, and thus have been widely adopted in classification tasks [23], [24], [25], [26], [27]. Such methods either directly use attributes to infer category or only use attributes as additional side information. Due to the usage of attributes, the above methods can explain their predictions to some extent. However, all these methods only use manually-defined attributes, which severely limit their scalability. Some recent works have shown that CNN models trained for classification tasks can naturally learn visual attributes as their intermediate representations [20], [21], [28], which provides an alternative way to significantly improve the scalability of attribute-based classification methods. However, these methods do not exploit the rich supervision information in class hierarchies. As a result, it is highly likely that the learned interpretations violate the hierarchical constraints as discussed in Section 1, making the interpretations less meaningful and reliable. Therefore, in this paper, we propose to incorporate the class hierarchy to learn more meaningful criteria that are in the same form as the hierarchical criteria described in Section 1.

Hierarchy-Regularized Classification. Semantic hierarchies have been explored in object classification task for accelerating recognition [29], [30], determining the appropriate model outputs [31], [32], making use of coarsely labeled images [33], [34], and improving recognition accuracy as additional supervision [14], [35], [36], [37], [38], [39]. While such methods have achieved their goals by leveraging

category hierarchies, only [14] pays attention to the task studied in this paper, i.e., learning the criteria for classification. However, [14] utilizes a small number of manually defined category-level attributes, which severely restricts the scalability of their method. To deal with this problem, we propose to jointly learn the attributes as well as the criteria for classification with only image-level class labels and a category hierarchy, which are much easier to acquire than category-level attribute labels.

Network Interpretation. Since the breakthrough of AlexNet [1] on the ImageNet visual recognition competition [40], various works have been proposed to interpret CNN models. Such methods make attempts to reveal the visual patterns that activate each convolutional filter [15], [18], [41], [42], [43], [44] or to locate the area in an image that is responsible for the prediction [19], [38], [45], [46], [47]. These methods have made a certain level of success in interpreting CNN models, however, they can only explain what has already been learned by the model, but cannot improve the model interpretability in return.

To deal with the aforementioned problem, some recent works propose to directly learn interpretable representations for classification. For instance, [17] proposes to explicitly constrain high-level CNN filters to represent object parts. [48] learns mid-level representations (prototypes) that resemble training inputs and can be decoded to images to obtain the typical visual appearance of each category. [49] improves the performance of the interpretable neural module network in the visual question answering task by enforcing the network to extract features from the attended image area. [16] explicitly aligns neural activations with human-nameable attributes to obtain interpretable classifiers for zero-shot learning task. However, all these methods ignore the rich hierarchical structure of categories and thus cannot learn the hierarchical interpretations as defined by human experts, which limits the naturalness of their interpretations. Moreover, without the constraints of class hierarchy, it is possible for such methods to learn contradictory interpretations. To address these issues, HCN makes use of the naturally existing class hierarchy to learn more human-friendly and meaningful interpretations.

3 APPROACH

As discussed in Section 1, attributes can inherently characterize the nature of object categories and the connections between them. The goal of this work is to learn such attribute-based classification criteria, i.e., the visual attributes that define the categories and how the categories are defined by these attributes. To achieve this goal, we borrow the classification mechanism from human experts by learning attributes and the hierarchical criteria based on such attributes in a data-driven manner, as shown in Fig. 2. To this end, a two-stream CNN architecture named Hierarchical Criteria Network is elaborately devised, where the two streams embed images and category prototypes into a common space respectively, as shown in Fig. 3. Specifically, the upper stream takes images as inputs and learns attributes (i.e., the convolution filters) as the bases of the embedding space. On the other hand, the lower stream is designed to learn the prototypes for the hierarchically organized

categories in the same space, so that each category could be represented by the linear combination of its superordinate category and some specific attributes associated with the considered category in this learned space. To ensure the meaningfulness of the learned criteria (as discussed in Section 1), a set of constraints are imposed on the prototypes to ensure the consistency between category-superordinate category pairs. To train the model, outputs from both streams are then fed into a shared classifier to predict the hierarchical structure of the corresponding samples.

Formally, HCN targets to learn an interpretable image classification model, where the interpretations produced by the model should be in a similar form as the hierarchical criteria defined by human experts. Suppose that we are given a set of Q images $X = \{\mathbf{x}_q | q \in 1, 2, \dots, Q\}$ that belong to N hierarchically organized categories $C = \{c_i | i \in 1, 2, \dots, N\}$, we would like to learn two distinctive mapping functions $f: X \rightarrow [0, \infty)^D$ and $g: C \rightarrow [0, \infty)^D$ to embed the images and category prototypes into a common D -dimensional space as shown in Fig. 3, where the images are aligned with their corresponding prototypes. In the following, we use boldface capital letters, e.g., \mathbf{A} , to denote matrix, where $\mathbf{A}_{i,}$, $\mathbf{A}_{,j}$, and $\mathbf{A}_{i,j}$ denote the i th row, j th column, (i, j) th element of \mathbf{A} respectively. The designation of the two mapping functions as well as how to train the model will be detailed in the following subsections.

3.1 Image Attributes

The upper stream (mapping function f) in HCN serves two purposes, i.e., learning the bases of the feature space and embed the images into this space. Specifically, this stream consists of a standard CNN model which takes an image as input and produces D feature maps (each corresponds to an attribute) of size $w \times h$ after a series of convolution, pooling, and non-linear activation operations, where w and h are the width and height of the output feature maps respectively, and the values in each feature map indicate the attribute strength at the specific location, which will be illustrated in the experiment part (Section 4.3). Since the features represent the attribute strength, we enforce them to have non-negative values, which can be easily achieved with ReLU activation functions [50]. Then the D attributes of the q th image in the training set $\mathbf{a}_q \in [0, \infty)^{1 \times D}$ is obtained by globally max pooling the D feature maps, where zero entries mean that the corresponding attribute is not presented in the image throughout the paper. Here we adopt global max pooling instead of the more generally adopted average pooling [2], [51] to intentionally concentrate only on the discriminative areas of the image and avoid interference from other areas. Note that in our definition, an attribute could either be “has a certain property” or “does not have a certain property”, and thus can naturally deal with the case where “does not have a certain attribute” is the key property of an object.

3.2 Category Prototypes

In this part, we explain the details of the lower stream (mapping function g), which is the key component that makes the HCN different from existing methods, in Fig. 3. Although different objects of the same category or even different views

of the same object could have drastic appearance variances, there are indeed some common attributes that group them together. The certain combination of attributes, which are also known as prototypes, could be seen as the criteria for classification. Suppose that the category hierarchy can be represented as a directed acyclic graph $G = \{C, E\}$, where the node $c_i \in C$ denotes the i th category as discussed above, and the edge $e_{j,i} \in E$ denotes that c_j is a superordinate category of c_i (either direct or indirect). Assume that there are N categories in total, the hierarchical representations of all categories $\mathbf{H} \in \{0, 1\}^{N \times N}$ can be defined as follows:

$$\mathbf{H}_{j,i} = \begin{cases} 1, & e_{j,i} \in E \text{ or } i = j \\ 0, & \text{otherwise} \end{cases}, \quad (1)$$

where \mathbf{H}_i represents the hierarchical structure of the corresponding category c_i . Moreover, let $\mathbf{P} \in [0, \infty)^{N \times D}$ be the prototypes of all categories, where D is the number of visual attributes as mentioned above, and each row in \mathbf{P} corresponds to a category in the hierarchy. With this definition, each entry in \mathbf{P} denotes the strength of an attribute on a certain category. Note that the zero entries in a row mean that the corresponding category does not have certain attributes, and the row of a category contains all its superordinate category's non-zero entries but has more non-zero entries, which are the specific attributes of that category. The training objective is to learn the values in \mathbf{P} such that they could faithfully reflect the strength of different attributes on the hierarchically organized categories.

As discussed in Section 1, our basic intuition is that each category inherits all attributes from its superordinate categories, and also has additional specific attributes that distinguish them from their superordinate categories. Therefore the values in \mathbf{P} should satisfy the following constraints:

$$\mathbf{P}_{i,k} \geq \mathbf{P}_{j,k} \text{ for } e_{j,i} \in E, \forall k \in \{1, \dots, D\}. \quad (2)$$

The constraints in Eqn. (2) can ensure that the learned prototypes are meaningful, i.e., no contradictions between any possible category-superordinate category pairs. While it is rather difficult to directly optimize the prototypes \mathbf{P} due to the complex constraints, we resort to optimizing an alternative embedding matrix \mathbf{M} with the help of the hierarchical representation \mathbf{H} , and rewrite \mathbf{P} as follows:

$$\begin{aligned} \mathbf{P} &= \mathbf{H}^T \mathbf{M} \\ \text{s.t. } \mathbf{M} &\in [0, \infty)^{N \times D}. \end{aligned} \quad (3)$$

To better understand the above definition, let us take a closer look at the simplified category hierarchy in the bottom left part of Fig. 3. As illustrated in the figure, we exemplify the two columns of \mathbf{H} that correspond to “feline” and “tabby” in red (left) and green (right) respectively as an example (the other columns can be obtained similarly). Each dimension of the hierarchical category representation indicates whether the corresponding category is on the path from the root node (i.e., *animal*) of the hierarchy to the category of “feline” and “tabby”. With the definition of \mathbf{H} in Eqn. (1), Eqn. (3) is equivalent to selecting and summing up the rows of \mathbf{M} under the instruction of \mathbf{H} as shown in the lower middle part of Fig. 3. Therefore, each row of \mathbf{M} can

be interpreted as the differences between a category and its direct superordinate category. With such a framework, HCN is able to derive the criteria for classification that resemble the expert-defined ones. To be specific, by computing the difference vector $\mathbf{d} = \mathbf{P}_{i \cdot} - \mathbf{P}_{j \cdot}$, $e_{j,i} \in E$, we could explicitly tell that the i th category is a special case of the j th category with the most significant attributes in \mathbf{d} .

3.3 Optimization

Given the above network structure designed for our criteria learning task, in this subsection, we describe how to optimize the model to align the two streams for learning the hierarchical classification criteria.

To train the model, both the image samples from the upper stream and the category prototypes from the lower stream are used as inputs to a shared classifier, as shown in the rightmost part of Fig. 3. Note that unlike the conventional CNN models that only learn to predict the leaf node categories, our classifier is designed to predict the whole path in the category hierarchy for each sample since the interpretations we are seeking for is also hierarchical. To be specific, one possible version of our classifier consists of two parts, where the first part is for the leaf node categories and the second part is for the inner node categories, as shown in the rightmost part of Fig. 3. Let $C_l \subset C$ denote the subset of all leaf node categories and $|C_l|$ is the number of leaf node categories. $\mathbf{W} \in \mathbb{R}^{D \times N}$ denotes the classifier weights shared by the two streams, where each column of \mathbf{W} corresponds to a category in the class hierarchy. Without loss of generality, the bias terms are ignored for simplicity, and we assume that for $i \in \{1, 2, \dots, N - |C_l|\}$ and $i > N - |C_l|$, c_i corresponds to an inner/a leaf node category respectively.

For the leaf nodes, as they are mutually exclusive, a softmax classification loss (e.g., softmax) can be adopted, which is denoted as L_{leaf} . The inputs to this part include all image samples and the prototypes of the leaf node categories (note that the other prototypes do not have leaf-level labels, and thus are not used in this part). Take the q th image sample \mathbf{x}_q as an example, the corresponding leaf-level loss is defined based on its attribute representation \mathbf{a}_q as follows:

$$L_{leaf}(\mathbf{a}_q) = -\log\left(\frac{e^{\mathbf{a}_q \mathbf{W}_{y_q}}}{\sum_{j=N-|C_l|+1}^N e^{\mathbf{a}_q \mathbf{W}_{j \cdot}}}\right), \quad (4)$$

where $y_q \in \{N - |C_l| + 1, \dots, N\}$ denotes the leaf category index of \mathbf{x}_q .

As for the other part, since the inner nodes are non-exclusive, a set of independent loss functions (e.g., sigmoid cross entropy loss) can be exploited, which is denoted as L_{inner} . For this part, all image samples and prototypes are used as inputs to predict the inner nodes they belong to. The inner-node loss for \mathbf{a}_q is defined as

$$L_{inner}(\mathbf{a}_q) = -\sum_{j=1}^{N-|C_l|} [\mathbf{H}_{j,y_q} \log(S(\mathbf{a}_q \mathbf{W}_{j \cdot})) + (1 - \mathbf{H}_{j,y_q}) \log(1 - S(\mathbf{a}_q \mathbf{W}_{j \cdot}))], \quad (5)$$

where $S(\cdot)$ is the sigmoid function.

Similarly, by replacing \mathbf{a}_q and y_q with $\mathbf{P}_{i \cdot}$ and i respectively, we can get $L_{leaf}(\mathbf{P}_{i \cdot})$ and $L_{inner}(\mathbf{P}_{i \cdot})$ as follows:

$$L_{leaf}(\mathbf{P}_{i \cdot}) = -\log\left(\frac{e^{\mathbf{P}_{i \cdot} \mathbf{W}_i}}{\sum_{j=N-|C_l|+1}^N e^{\mathbf{P}_{i \cdot} \mathbf{W}_{j \cdot}}}\right), \quad i > N - |C_l|, \quad (6)$$

$$L_{inner}(\mathbf{P}_{i \cdot}) = -\sum_{j=1}^{N-|C_l|} [\mathbf{H}_{j,i} \log(S(\mathbf{P}_{i \cdot} \mathbf{W}_{j \cdot})) + (1 - \mathbf{H}_{j,i}) \log(1 - S(\mathbf{P}_{i \cdot} \mathbf{W}_{j \cdot}))]. \quad (7)$$

In summary, the overall loss function can be written as follows:

$$L = \alpha \sum_q L_{leaf}(\mathbf{a}_q) + \beta \sum_{c_i \in C_l} L_{leaf}(\mathbf{P}_{i \cdot}) + \gamma \sum_q L_{inner}(\mathbf{a}_q) + \delta \sum_{c_i \in C} L_{inner}(\mathbf{P}_{i \cdot}). \quad (8)$$

where $\alpha, \beta, \gamma, \delta$ are weighting parameters for different parts of the overall loss. Note that by setting $\alpha = 1, \beta = \gamma = \delta = 0$, HCN reduces to standard classification model. While here we adopt relatively simple forms of loss functions, other more sophisticated hierarchical loss functions (e.g., the graph model in [33]) are also compatible with HCN.

To optimize with the non-negative constraints on the embedding matrix \mathbf{M} , Eqn. (3) is relaxed as follows:

$$\mathbf{P} = \mathbf{H}^T r(\mathbf{M}_u), \quad (9)$$

where \mathbf{M}_u is the unconstrained version of \mathbf{M} , $r(\cdot)$ is a derivable function with non-negative outputs (e.g., ReLU) and is applied element-wisely. With the above definitions, the model parameters (convolution filter parameters in the upper stream, \mathbf{M}_u in the lower stream, and the classifiers weights \mathbf{W}) can be optimized with standard back-propagation algorithm.

3.4 Human Feedback for Model Correction

Since the criteria learned by HCN is in the same form as the expert-defined criteria, it favorably enables the possibility of incorporating human feedback to further correct the models. Specifically, given an already trained model, by comparing the criteria learned by the model and the expert-defined ones, we can figure out what attributes are absent in the current model and what irrelevant attributes are unintentionally considered important by the model. By ignoring the irrelevant attributes and additionally incurring missing relevant attributes, the model can be adjusted according to external human knowledge.

For the first case, here we describe one simple yet feasible way of choosing the attributes to ignore for each category, while other selection schemes can also be possible. To be specific, in the leaf-level classifier, we use $\mathbf{v} = \mathbf{P}_{i \cdot}^T \otimes \mathbf{W}_i$ to denote the element-wise productions of the i th category prototype and the i th classifier weights, where $\mathbf{v} \in R^D$ and \otimes denotes the element-wise product of two vectors. The values in \mathbf{v} indicate the contribution of the attributes to the i th category, i.e., the larger \mathbf{v}_k is, the more important the k th attribute is to the i th category. Therefore, a straightforward scheme is to first sort the values in \mathbf{v} and select some of the least important attributes to be removed and then set the corresponding values in \mathbf{W}_i to zero. After that, the

TABLE 1
The Network Structure of the Upper Stream
on the CIFAR-100 Dataset

Layer type	Parameters*	Output size
Convolution + BN + ReLU	$5 \times 5 \times 64$, stride 1	$32 \times 32 \times 64$
Max Pooling	3×3 , stride 2	$16 \times 16 \times 64$
Convolution + BN + ReLU	$5 \times 5 \times 64$, stride 1	$16 \times 16 \times 64$
Average Pooling	3×3 , stride 2	$8 \times 8 \times 64$
Convolution + BN + ReLU	$5 \times 5 \times 64$, stride 1	$8 \times 8 \times 64$
Average Pooling	3×3 , stride 2	$4 \times 4 \times 64$
Convolution	$1 \times 1 \times D$, stride 1	$4 \times 4 \times D$
Max Pooling	4×4 , stride 1	$1 \times 1 \times D$
Leaky ReLU	negative slope 0.1	$1 \times 1 \times D$
Fully Connected	121 nodes	$1 \times 1 \times 121$

* The second column lists the parameters (e.g., kernel size size (\times filter number)) of the corresponding layers, where D is the number of attributes as described in the main paper.

unselected values of \mathbf{W} are finetuned, while convolutional filters in the upper stream and the embedding matrix \mathbf{M} in the lower stream are kept fixed to avoid changing the meaning of the attributes.

For the second case, one possible feedback scheme is to add those missing important attributes to the model (e.g., adding a “red cross” attribute for *ambulance*), which could be achieved by appending a new filter to the attribute layer and finetuning the model with an additional “red cross” prediction loss on the newly incorporated filter. Such a scheme would surely require some expert knowledge and additional labeling workload, but compared to previous classification methods that are purely based on manually labeled attributes [22], [24], HCN only needs to label a minor portion of attributes that are not captured by the model, while the large portion has already been automatically learned. We will evaluate the above schemes as well as some alternative ones in detail in Section 4.7 to validate the feasibility of manually removing irrelevant or adding missing attributes in HCN.

4 EXPERIMENTS

In this section, we extensively evaluate HCN on two widely studied datasets. We first evaluate HCN by varying its different modules. Then a series of qualitative and quantitative experiments are conducted to show the effectiveness of HCN. Finally, we compare to previous methods to demonstrate the superiority of HCN.

4.1 Experimental Settings

Datasets. Two hierarchically organized datasets are considered in our experiments: (1) *CIFAR-100* [52] consists of 60,000 32×32 images belonging to 100 mutually exclusive categories (600 images per category), and the 100 categories are uniformly divided into 20 superclasses. Considering that all these superclasses belong to the category *entity*, we add a root node to the hierarchy, resulting in a hierarchy of 121 nodes. (2) *ILSVRC* [40] is a subset of the ImageNet dataset with 1,000 leaf node categories, and the whole category hierarchy consists of 1,860 nodes. Note that unlike the *CIFAR-100* dataset, in this hierarchy, some categories have

TABLE 2
The Network Structure of the Upper Stream on the
ILSVRC Dataset, Which is a Slightly Modified Network
in Network (NIN) Model

Layer type	Parameters	Output size
Convolution + ReLU	$11 \times 11 \times 96$, stride 4	$54 \times 54 \times 96$
Convolution + ReLU	$1 \times 1 \times 96$, stride 1	$54 \times 54 \times 96$
Convolution + ReLU	$1 \times 1 \times 96$, stride 1	$54 \times 54 \times 96$
Max Pooling	3×3 , stride 2	$27 \times 27 \times 96$
Convolution + ReLU	$5 \times 5 \times 256$, stride 1	$27 \times 27 \times 256$
Convolution + ReLU	$1 \times 1 \times 256$, stride 1	$27 \times 27 \times 256$
Convolution + ReLU	$1 \times 1 \times 256$, stride 1	$27 \times 27 \times 256$
Max Pooling	3×3 , stride 2	$13 \times 13 \times 256$
Convolution + ReLU	$3 \times 3 \times 384$, stride 1	$13 \times 13 \times 384$
Convolution + ReLU	$1 \times 1 \times 384$, stride 1	$13 \times 13 \times 384$
Convolution + ReLU	$1 \times 1 \times 384$, stride 1	$13 \times 13 \times 384$
Max Pooling	3×3 , stride 2	$6 \times 6 \times 384$
Convolution + ReLU	$3 \times 3 \times 1024$, stride 1	$6 \times 6 \times 1024$
Convolution	$1 \times 1 \times D$, stride 1	$6 \times 6 \times D$
Max Pooling	6×6 , stride 1	$1 \times 1 \times D$
Leaky ReLU	negative slope 0.1	$1 \times 1 \times D$
Fully Connected	1,860 nodes	$1 \times 1 \times 1860$

more than one direct superordinate categories, making this dataset more challenging. On both datasets, we follow the standard data partition protocol to train and test HCN.

Implementation Details. HCN is implemented with the open-source toolbox Caffe¹ [53]. The upper stream of HCN consists of several convolutional, pooling, and non-linear activation layers. The architectures of the upper stream on both datasets are detailed in Tables 1 and 2. On the other hand, the lower stream is implemented by a single inner product layer, as shown in Fig. 4. For *CIFAR-100* dataset, the resolution of input images is 32×32 . Considering the small image resolution, we use a simple network of four convolution layers with batch normalization (BN) [54] and ReLU activation before the global max pooling operation. As for *ILSVRC*, the images are resized to 256×256 , and 224×224 patches are randomly cropped as inputs. Since this is a much more complicated dataset, we adopt a slightly modified Network In Network (NIN) structure [51] for speed-accuracy trade-off, where we change the order of global pooling to better suit the problem at hand. Note that HCN is not limited to the network structures described above, more advanced structures (e.g., ResNet [5]) are also compatible with HCN. However, training such deep architectures would take much more time,² and thus makes it difficult to conduct thorough evaluations. In our experiments, the model parameters are randomly initialized with the method in [55] and trained from scratch using SGD algorithm with a momentum of 0.9.

As a widely exploited activation function, ReLU could be adopted to impose the non-negative constraints in both streams (after the global max pooling operation in the upper

1. The source code of HCN with running samples are available at <http://vip.ict.ac.cn/resources/codes>

2. In our experiments, training NIN from scratch takes only 1 day on a single GPU, while training a commonly used 50-layer ResNet model takes about two weeks on 2 GPUs.

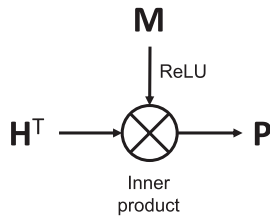


Fig. 4. The structure of the lower stream of HCN.

stream and $r(\cdot)$ operation in the lower stream). However, in practice, we have found that using ReLU activation results in about 1/3 of “dead” attributes, i.e., some dimensions of the attribute vectors are always zero for all images. To deal with this problem, we resort to leaky ReLU [56], which has a small negative slope, as an alternative. While using such an activation function could keep all attributes active, it would inevitably violate the non-negative constraints on the image attribute features \mathbf{a} and the prototypes \mathbf{P} . Nonetheless, in our experiments, it is noted that the negative values are negligible compared to the positive ones (e.g., even the most negative value in $r(\mathbf{M}_u)$ is only -4.5×10^{-4}). Therefore, using the leaky ReLU has more pros than cons, and we will use this alternative in the following experiments.

4.2 Module Analysis

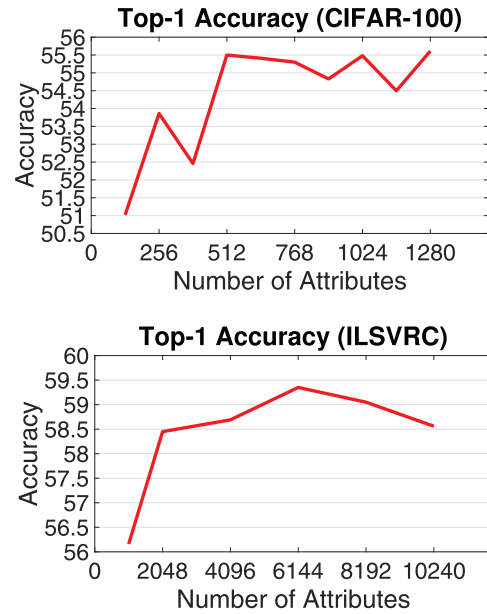
4.2.1 Weighting Parameters

For CIFAR-100 dataset, we heuristically set the number of attributes $D = 128$. As for the ILSVRC dataset, following the standard network architecture of NIN, we use $D = 1,024$ attributes in these experiments. Table 3 shows the leaf-level classification accuracies of HCN under different weighting parameter settings, where γ, δ are set with grid search to make sure that HCN can achieve satisfactory classification accuracy and α, β are set in a heuristic manner. The top and bottom panels in Table 3 show the results with different weighting parameters on CIFAR-100 and ILSVRC respectively. It can be observed that: *First*, incorporating the inner node loss ($\gamma, \delta > 0$) degrades the leaf-level classification accuracy in most cases, which might be caused by the increased difficulty of training target, i.e., the model is expected to learn not only the leaf-level concepts but also some more abstract concepts (the inner node concepts). In fact, as will be demonstrated in the following Section 4.2.2,

TABLE 3
Accuracies With Different Weighting Parameter Settings

α	β	γ	δ	Top-1 accuracy	Top-5 accuracy
1	0	0	0	52.00%	81.32%
0.5	0.5	0	0	53.61%	82.78%
0.98	0	0.02	0	51.73%	82.10%
0.49	0.49	0.01	0.01	50.63%	81.17%
0.69	0.29	0.01	0.01	51.02%	81.11%
1	0	0	0	57.12%	80.01%
0.5	0.5	0	0	54.02%	77.84%
0.998	0	0.002	0	56.86%	79.78%
0.499	0.499	0.001	0.001	55.04%	78.55%
0.699	0.299	0.001	0.001	56.16%	79.35%

The top and bottom panels correspond to CIFAR-100 ($D = 128$) and ILSVRC ($D = 1,024$), respectively.


 Fig. 5. Top-1 accuracy (%) on the two datasets with different number of attributes D .

when $D = 6,144$ (on the ILSVRC dataset), HCN surpasses the baseline model and even the standard NIN model. A possible explanation is that the added task requires more model capacity. When the number of attributes is small, the model capacity is not large enough to jointly deal with the multiple tasks, and thus the model has to balance the different tasks, resulting in a performance drop of the leaf-level classification task. As the number of attributes increases, the model capacity also increases, and thus the model is able to well handle the multiple tasks simultaneously and the performance of leaf-level classification task increases accordingly, where the loss for predicting inner nodes of the hierarchy could serve as a regularizer for the leaf-level classification task. Note that the baseline model is purely optimized for accuracy and incapable of learning the category prototypes and hierarchical classification criteria, since the lower stream has no supervision signal. Such comparison suggests that HCN could improve the model interpretability with little or no performance degradation. *Second*, it is better to use larger weights for the upper stream (4th versus 5th rows in each panel) when training both streams. This phenomenon can be explained by the fact that the number of prototypes in the lower stream is much smaller than the number of training images in the upper stream. As a result, assigning large weights to the lower stream might cause overfitting and thus degrades the accuracy.

Considering the classification performance of the compared models, we will set $\alpha = 0.69$, $\beta = 0.29$, $\gamma = \delta = 0.01$ on CIFAR-100 and $\alpha = 0.699$, $\beta = 0.299$, $\gamma = \delta = 0.001$ on ILSVRC in the following experiments, and we will refer to the model with the weighting parameters $\alpha = 1, \beta = \gamma = \delta = 0$ as the baseline model unless otherwise specified.

4.2.2 Number of Attributes

With the above weighting parameters, we evaluate the leaf-level classification accuracy of HCN under different numbers of attributes. Fig. 5 shows the top-1 classification accuracy (%)

of HCN with varying numbers of attributes. We can see that the best numbers of attributes in terms of accuracy (55.50/84.23 percent and 59.35/81.32 percent top-1/5 accuracy) are 512 and 6,144 for the two datasets respectively. For comparison, the baseline model with $D = 6,144$ achieves 58.71/80.65 percent top-1/5 accuracy on the ILSVRC validation set, and the standard NIN model released in the model zoo of Caffe³ achieves 59.11/81.32 percent top-1/5 accuracy. Such results suggest that HCN is able to achieve comparable or even better performance than the baseline model and the standard classification models once the model capacity is large enough, validating the advantage of the HCN.

On the ILSVRC dataset, when the number of attributes is larger than 6,144, the model performance starts to drop. On the contrary, there is no significant performance drop when increasing the number of attributes on CIFAR-100. A possible explanation is that as the number of attribute increases, the model has more parameters and is easier to overfit (which is the case on ILSVRC dataset), while the batch normalization operation in the CIFAR-100 models can relieve this problem (the backbone model we adopted on ILSVRC does not use batch normalization). Based on the above results, we finally set the number of attributes $D = 512$ and $D = 6,144$ on CIFAR-100 and ILSVRC respectively in the following experiments unless otherwise specified.

Here we make an attempt to roughly estimate the number of attributes that are required to classify the 1,000 ILSVRC categories. For reference, let us take a look at some widely studied existing object classification datasets with attribute labels. The Caltech-UCSD Birds dataset [57] aims at classifying 200 species of birds, and 312 visual attributes are manually labeled on this dataset. Animals with Attribute (AWA) dataset [24] and aPY dataset [22] are widely adopted in zero-shot learning tasks, which have 50 / 32 categories and 85 / 64 attributes respectively. COCO Attributes [58] is one of the largest object attribute datasets, and it contains 29 object categories and 196 attributes. There are also some other attribute datasets, e.g., CelebA [59] and SUN Attribute [60], however, they concentrate on other tasks (i.e., face recognition and scene recognition) instead of general object recognition. Therefore, we do not use them as references.

It can be easily seen from the above statistics that the number of manually-labeled attributes is usually at least 1.5 times as large as the number of categories. Therefore, we could confidently assume that at least 1,500 human-defined attributes are required to well distinguish the 1,000 ILSVRC categories. More importantly, since there are many fine-grained categories in ILSVRC (e.g., about 100 breeds of dogs), a large portion of such attributes need to be defined and labeled by human experts instead of ordinary persons. Therefore, labeling such a large amount of attributes for large-scale object recognition task could be very expensive. In contrast, HCN is able to learn such attributes automatically, and thus is more scalable than manually defining attributes.

4.2.3 Architecture of Upper Stream

In this part, we show that by using the more advanced ResNet-18 [5] as the backbone architecture for the upper

TABLE 4
Top-1/5 Accuracy of the Standard ResNet-18 Model, the Baseline Model With ResNet-18 Backbone, and HCN Model With ResNet-18 Backbone on the ILSVRC Validation Set

	Standard [5]	Baseline ($\alpha = 1$)	HCN
Top-1 accuracy	67.57%	68.92%	68.77%
Top-5 accuracy	88.10%	88.84%	88.94%

stream (due to the limited computation resource, we are unable to train deeper networks on our machine, such as ResNet-50). The CNN architecture is slightly modified by adding another convolution layer between the last residual block and the global pooling layer. Specifically, the added layer contains 6,144 convolution kernels of size 1×1 to produce 6,144 attributes, since HCN has the best performance with 6,144 attributes in the previous experiments.

We compare three models: (1) the standard ResNet-18 model as proposed in [5]. Since the official ResNet-18 model of [5] is not released to the public and the original publication does not report the results under the same test settings as ours (i.e., single crop test, and both sides of the images are resized to 256). For a fair comparison, in this experiment, the results of the standard ResNet-18 model are obtained with the non-official implementation⁴ under the same test settings as in our experiments, which achieves comparable performance with the results reported in [5]. This model serves as a representative state-of-the-art classification method; (2) the baseline model with ResNet-18 architecture. This model corresponds to the first row in the bottom part of Table 3, and is used to show the performance of the modified network structure when trained purely on the 1,000-way classification task; (3) HCN with ResNet-18 architecture as the upper stream, which corresponds to the last row in Table 3.

The classification accuracies of the compared models on the validation set of ILSVRC are shown in Table 4. The results suggest that HCN can undoubtedly achieve state-of-the-art performance with more advanced backbone architecture. Moreover, we can confidently assume that with even deeper and more advanced architecture, e.g., DenseNet [7], the classification accuracy can be further improved. However, due to the huge computational overhead of such models, we will still use NIN as the backbone architecture for HCN in the following experiments.

4.2.4 Choice of Non-Negative Constraint Function

In the above experiments, leaky ReLU is adopted to impose the non-negative constraints on M . On one hand, the leaky ReLU might violate the non-negative constraint, while on the other hand, the standard ReLU results in a large portion of dead attributes as discussed above in Section 4.1. In this part, we make an attempt to evaluate the possibility of imposing the non-negative constraints with another widely adopted non-linear functions by replacing the leaky ReLU activation function with the Sigmoid activation (with batch normalization to ensure convergence). After carefully tuning the learning rate schedule, such model still achieves

3. <https://github.com/BVLC/caffe/wiki/Model-Zoo>

4. <https://github.com/HolmesShuan/ResNet-18-Caffemodel-on-ImageNet>

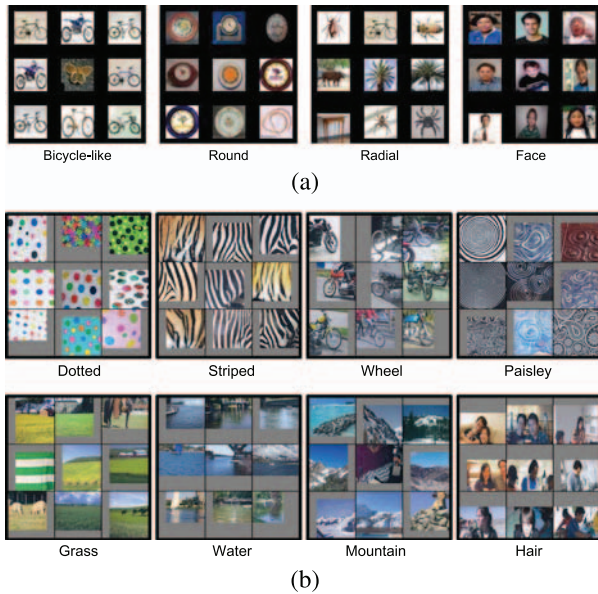


Fig. 6. Some attributes learned by HCN on (a) CIFAR-100 and (b) ILSVRC, respectively. For each attribute, we show the 9 image patches that maximally activate the corresponding convolution filter, and give the meanings of these attributes below the images. For CIFAR-100, we give our interpretations of these attributes, and for the ILSVRC dataset, the concepts correspond to these attributes are generated by [18].

very poor performance. Specifically, the Sigmoid model achieves 49.22/73.77 percent top-1/5 accuracy on the ILSVRC validation set with 1,024 attributes, which is about 6 percent lower than the leaky ReLU model in Table 3. A possible explanation is that the outputs of Sigmoid function are limited to the range $(0, 1)$ while the outputs of leaky ReLU is in the range of $(-\infty, \infty)$ in theory (in our experiments, the range is usually $(-0.001, \infty)$), and thus the Sigmoid outputs have smaller L2 norm than the outputs of ReLU function, which results in worse performance [61]. To be more precise, the authors of [61] suggests that the features with a small norm reside on a hypersphere with small surface area, and such a limited surface area is not large enough for embedding features from the same class together and those from different classes far from each other. There might be some more proper functions for imposing the non-negative constraints, however, that is beyond the scope of this paper.

To summarize, the above results validate that the leaky ReLU function has more pros than cons as the choice of non-negative constraint function, and thus we will use the leaky ReLU in the following experiments.

4.3 Learned Attributes

The convolution filters in the last convolution layer of HCN act as attribute extractors. To visualize the learned attributes, we use the toolboxes of [62] to find the 9 image patches with the largest activation values on each of the D filters to represent the attributes. Specifically, since the images on CIFAR-100 are very small and are quite different from natural images, we compute the top activations on the training set of CIFAR-100 to give the meanings of the attributes based on our understanding. On the other hand, since ILSVRC dataset is biased towards dogs (i.e., there are about 100 out of 1,000 categories of dogs in ILSVRC dataset), we

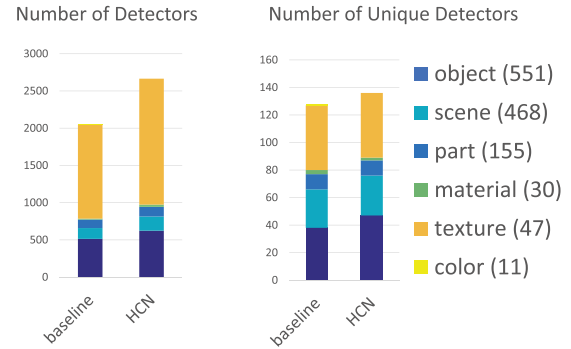


Fig. 7. The number of detectors and the number of unique detectors computed using the source codes and dataset proposed by network dissection [18]. The baseline model ($\alpha = 1$) and HCN with $D = 6,144$ attributes are evaluated.

use the less biased BRODEN dataset [18] to compute the top activations. The BRODEN dataset has pixel-level annotations of 1,262 well-annotated concepts (6 general types) on 63,305 images, and [18] computes the correspondence between each convolutional filter and the labeled concepts to quantize the interpretability of individual convolutional filters, i.e., the learned attributes in HCN, and give the meaning of each attribute. Specifically, if the top activations of a convolutional filter strongly correlate with a concept, the corresponding convolutional filter is deemed as a detector of that concept, which indicates the name of the corresponding attribute.

Some randomly selected attributes on both datasets along with their meanings are shown in Fig. 6 (note that only in this figure, the meanings of attributes on ILSVRC are given by [18]). We can observe that the attributes learned on CIFAR-100 are mainly describing the shape of objects, while the attributes on ILSVRC are more diverse, including textures, object parts, and even high-level semantic attributes (e.g., water and mountain). This is in line with what we have expected since the images in CIFAR-100 have much smaller resolutions compared to the images in ILSVRC, and thus detailed visual information such as texture can hardly be learned from this dataset. Moreover, since the category hierarchy on ILSVRC is much more complicated, the model on this dataset has to learn attributes at different abstraction levels to well distinguish these categories.

To quantitatively evaluate the interpretability of the learned attributes, we use the publicly available source codes and dataset released by [18] to compute the number of detectors (interpretable attributes) and the number of unique detectors. We compare the baseline model and the proposed HCN model with $D = 6,144$ attributes. The results are shown in Fig. 7, where the total numbers of concepts of each general type are given in the brackets and they sum up to 1,262. We can see that HCN learns more interpretable attributes (detectors) and more unique detectors than the baseline model, validating the advantage of HCN in terms of interpretability. Moreover, Fig. 8 shows some attributes that are considered as uninterpretable by [18]. However, they could possibly be interpreted as “bird” and “dark dot on white background” respectively. The above results seem to suggest that even the current state-of-the-art method might fail to find the correspondence between filters and concepts in some cases (e.g., the first attribute in



Fig. 8. Examples of some learned attributes that are considered as “uninterpretable” by the visualization tool of network dissection [18].

Fig. 8), and more concepts are needed to evaluate the interpretability of CNN models (e.g., the second attribute in Fig. 8). Moreover, as indicated by [44], an attribute might be jointly encoded by multiple filters, and thus more appropriate methods are required for objectively evaluating the degree of interpretability. Therefore, quantitatively evaluating the interpretability of CNN models still remains an open problem.

To further validate that the learned attributes are shared among categories, for each attribute, we find the leaf node categories in ILSVRC to which the attribute contributes most. Specifically, since the leaf node categories have different depths in the hierarchy and the corresponding prototypes are summed over the whole path, the absolute values in the prototype vectors are affected by the depth and may not be a proper indicator of the actual strength of the attribute across all categories. To deal with this problem, we define a normalized indicator instead. For the k th attribute, we find the five categories with the largest $(\mathbf{P}_{i,k} \cdot \mathbf{W}_{k,i}) / (\mathbf{P}_i \cdot \mathbf{W}_i), \forall i$, which is the relative contribution of the selected attribute to the i th category, and some of the results are shown in Fig. 9. As can be seen, the attributes shown in the leftmost part of Fig. 9 are shared among very different categories rather than just fitting to a single category, validating the generality of the learned attribute. Moreover, some heatmaps of the two selected attributes on real images are shown in Fig. 10, which further validates the correspondence of the attribute with sharp/spotty parts in the images.

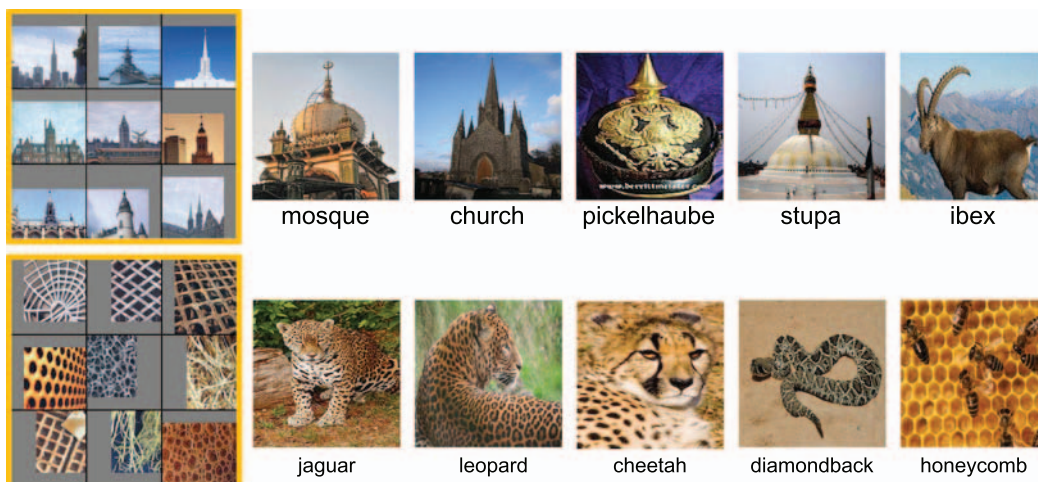


Fig. 9. Visualization of the five leaf node categories in ILSVRC to which the attributes shown in the leftmost block (sharp and spotty) contribute most.



Fig. 10. The heatmaps of activations on two attributes (filters). The top row shows the original images and their activation heatmaps on the attribute “sharp,” and the bottom row corresponds to the attribute “spotty.” In the heatmaps, warm color indicates high activations and cold colors indicate low activations. The attributes here have high activations at the expected parts in the images and low activations elsewhere, even though the highly activated areas correspond to different object categories. This figure is best viewed in color.

4.4 Comparison With Attribute-Based Classification Methods

In this part, we compare the classification performance of the learned attributes with existing attribute-based classification methods. There are several existing datasets with attribute annotations [22], [24], [57], [58], [59], [60], [64], [65]. Among these datasets, [64] and [65] are based on the ImageNet dataset and the ILSVRC dataset respectively, which are similar to the dataset used in our experiments. However, the official website of ImageNet is currently under maintenance and thus the original images of [64] cannot be downloaded from the

TABLE 5
Top-1 Classification Accuracy on ImageNet-150K
Dataset With Different Attributes as Features

	(F1) Manually-labeled attributes	(F2) Learned attributes	(F1-p) Predicted attributes
Top-1 accuracy	10.35%	59.60%	6.85%
	(F3) DBC [63]	(F4) Combined (groundtruth)	(F5) Combined (predicted)
Top-1 accuracy	31.70%	61.55%	26.35%

ImageNet website. Therefore, we choose to use the ImageNet-150K dataset in [65] in this experiment. This dataset has 150,000 images of 1,000 ILSVRC categories, and 50,000 out of 150,000 images are labeled with 25 attributes, including color, shape, texture, material, and object parts, which are quite similar to the 25 attributes in [64]. Since the goal of this experiment is to compare with attribute-based classification methods, we only use the 50,000 images with attribute annotations in this experiment. Specifically, we use the official training images (48,000 images) to train a multi-class linear SVM classifier and the rest 2,000 images are used to test the performance of the classifier.

To train the SVM classifier, in this part, we compare four kinds of input features extracted from each image: (F1) 25-D manually-labeled attributes, where each dimension takes the value from $\{1, 0, -1\}$, indicating whether the corresponding attribute is presented (1), unknown (0) or not presented (-1) in the corresponding image. This feature is designed to evaluate the effectiveness of classification with manually-labeled attributes, and here both the training and test images use the manually-labeled attributes values; (F2) 25-D learned attributes for the ImageNet-150K dataset using HCN. Specifically, we train an HCN model on ILSVRC dataset with $D = 25$ attributes (which achieves 38.91/64.55 percent top-1/5 accuracy on the ILSVRC validation set) and extract the 25-D attributes as features; (F1-p) 25-D predicted attributes. We train 25 independent attribute classifiers on the training set of ImageNet-150K to predict the manually-labeled attributes, where each image is represented by the preceding layer's outputs of the 25 learned attributes in the CNN model (i.e., the model of F2. Note that to reduce feature dimensionality, we use the 1,024-D averagely pooled CNN outputs instead of the original one, which is 36864-D, i.e., $1,024 \times 6 \times 6$. The 25 attribute classifiers achieve 96.17 percent mean accuracy on the validation set of ImageNet-150K), and use the predicted attributes on the whole ImageNet-150K dataset as features. This feature is designed to evaluate a more realistic setting, i.e., classification based on the predicted attributes instead of manually labeling all images; (F3) 25-D attributes learned by

Discriminative Binary Codes (DBC) [63] using the same feature inputs as F1-p. This feature is designed to validate the advantage of the proposed method over previous attribute learning method.

The classification accuracies of the four kinds of inputs are shown in Table 5. It is easy to find that the proposed method performs best among all features (F1, F2, F1-p, and F3), suggesting that the attributes learned by HCN are more suitable for object recognition task than manually-defined attributes and the attributes learned by the previous method (i.e., DBC [63]). Further explanations on the results are as follows: HCN jointly learns D attributes for the hierarchical classification task by exploiting category relationships, and thus can learn discriminative features (attributes). In contrast, the attributes of F1 are manually-defined, which is mainly intended to describe the objects rather than distinguishing them, and thus it is likely that the discriminative power of the defined attributes is not strong enough to distinguish the 1,000 categories. On the other hand, DBC learns the D attributes independently, which would inevitably result in redundancy between the attributes. Therefore, DBC performs inferior to HCN. Moreover, both F1 and F3 ignores the naturally existing hierarchical structure of categories, which more or less degrades the efficiency of their attributes.

4.5 Consistency Between Two Streams

In HCN, although the two streams are trained on the same task, there is actually no explicit constraint being imposed on the consistency between their representations. Therefore, one might naturally doubt how well the prototypes and the image samples are aligned with each other.

For this purpose, we randomly select 10 leaf node categories on ILSVRC and 300 images per category in the training set. Since we use softmax loss for the L_{leaf} classifier as described in Section 3, samples of the same category (either images or the corresponding category prototype) should spread over similar directions in the feature space, as indicated in [66]. We compute the cosine distances (i.e., relative directions, which is defined as $1 - \text{cosine similarity}$) between samples (3,000 image samples of the 10 leaf categories and all 1,000 leaf-level category prototypes) to quantitatively evaluate how well the two streams are aligned, and the results are listed in Table 6. We can see that: *First*, even though the model achieves 59.35 percent top-1 accuracy, the average within-class (same category, SC) cosine distance between image samples are still quite large (Image-SC, 0.7300) in the high dimensional embedding space, and the distances between image samples and their corresponding category prototypes (Proto-SC, 0.7530) are comparable to the within-class distances of images, suggesting that the image samples are in similar directions as their corresponding prototypes. *Second*, the image samples are much closer to their corresponding

TABLE 6
Average Cosine Distances Between the Selected Image Samples and Their Distances to the Prototypes

	Image-SC	Proto-SC	Proto-1NN	Proto-2NN	Proto-3NN	Proto-4NN	Proto-5NN	Proto-all
Image	0.7300	0.7530	0.7443	0.7475	0.7505	0.7524	0.7535	0.8340

Image-SC: images of the same category, Proto-SC: prototype of the corresponding same category with the image, Proto-kNN: kth nearest prototype of the image, and Proto-all: all 1,000 prototypes.
The results are obtained on ILSVRC.

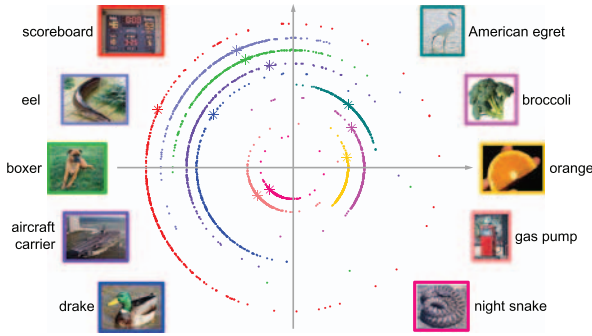


Fig. 11. Visualization of the image samples and the corresponding category prototypes in a two-dimensional space, where each color indicates a category, and the dots / asterisks (*) denote image samples and category prototypes respectively. Note that the points of each category are placed on concentric circles with different radii for clarity, and the radii do not have essential meanings.

prototypes than to irrelevant prototypes (Proto-SC versus Proto-all), and the distances from image samples to their corresponding prototypes are smaller than their distances to their 5th nearest prototypes (Proto-SC versus Proto-5NN), which further confirms that the image samples and their corresponding prototypes spread over the similar directions in the embedding space.

To gain an intuitive sense of the directions in the original space, we further propose to visualize the samples in a 2-D plane. Although the widely adopted visualization technique t-SNE [67] is able to preserve the cosine distances (i.e., directions) in the original space with the euclidean distance in the embedding space, it is suboptimal due to the discrepancy between the two types of distances. To deal with this problem, we propose a simple yet novel visualization technique to preserve the directions in the high dimensional space with the directions in a low dimensional space. To this end, we first normalize the D -dimensional representations of the selected image samples and their corresponding category prototypes to have unit norm so that their inner products are equivalent to their cosine similarities. After that, the normalized vector samples are projected into a two-dimensional space by performing PCA, which could preserve the inner products between samples in the original space. For clarity, the two-dimensional vectors from different categories are respectively scaled to have different norms so that they fall on a set of concentric circles with different radii and their directions in the low-dimensional space are preserved. The results are shown in Fig. 11, where each color corresponds to a category, the dots denote image samples, and the asterisks denote the category prototypes. Note that due to the enormous information loss when reducing feature dimension from 6,144 to 2, the directions in Fig. 11 are only rough approximations of the original ones, which should not be considered as a precise representation of the original directions. This figure shows that the image samples and the corresponding category prototypes spread over similar directions in most cases, which further validates that the two streams in HCN are well aligned with each other.

Moreover, we have also tested adding a loss term to enforce the consistency between the two streams on ILSVRC dataset. Specifically, we add a euclidean loss between the normalized image attribute vector and the corresponding category prototype vector as follows:

TABLE 7
The Classification Accuracy and Degree of Interpretability of HCN and the Modified Model With Explicit Constraints on the Consistency Between the Two Streams

	HCN	Modified model
Top-5 accuracy	81.98%	82.83%
Number of detectors	2,050	2,150
Number of unique detectors	128	108

$$L_{cons} = \left\| \frac{\mathbf{a}_q}{\|\mathbf{a}_q\|_2} - \frac{\mathbf{P}_{y_q}}{\|\mathbf{P}_{y_q}\|_2} \right\|_2^2, \quad (10)$$

where the subscript “cons” means consistency, and the other symbols have the same meaning as in Section 3.

By adding the above loss term to the overall loss Eqn. (8), the overall loss with consistency loss is as follows:

$$L_{all} = L + \mu L_{cons}, \quad (11)$$

where μ is a weighting parameter to control the relative importance of the consistency loss. We have carefully tuned μ in the range of $\{0.01, 0.1, 1.0, 10\}$, and the best leaf-level accuracy is obtained with $\mu = 1.0$. The performance of the original HCN, as well as the modified model (with 6,144 attributes), are listed in Table 7 (classification accuracy and number of detectors computed using [18]). Although the consistency loss can slightly improve the leaf-level accuracy and the number of detectors, it significantly decreases the number of unique detectors. A possible explanation is that the consistency loss acts similarly as the widely studied center loss [68], which could reduce the within-class variation and thus improve the classification performance. However, a side effect of the smaller within-class variation is that some variations are compressed, and thus the model learns fewer unique detectors. Since the main purpose of HCN is to improve the interpretability of the model, it is preferred not to add this loss term to the HCN framework.

4.6 Learned Criteria

The above experiments have validated that HCN can learn interpretable and discriminative attributes from images. In this part, we evaluate the classification criteria discovered by HCN on the two datasets.

First of all, we evaluate whether HCN has correctly learned the criteria for distinguishing a category from its superordinate category. For this purpose, we compute the D -dimensional difference vector between the prototypes as described in Section 3.2. The values in the difference vector \mathbf{d} are then sorted in descending order, and the attributes corresponding to the largest differences (i.e., the most important attributes for distinguishing the two categories) for some selected category pairs are shown in Fig. 12. Same as the above results, we denote each attribute by the top-9 activated images. We make three observations from the results: *First*, although some attributes in the learned criteria do not make much sense (e.g., the third attribute in the *ambulance* case), most of the criteria learned by HCN seem reasonable. For instance, *clock* is a “round” and “radial”



Fig. 12. Visualization of some learned criteria on CIFAR-100 (first row) and ILSVRC (last six rows) dataset, where five attributes with the largest differences are shown in each case to illustrate the key distinctions between the two categories on the left and right hand sides of the equal sign.

household electrical device; cheetah is a “spotted feline”; ambulance is a “light-colored (1st and 2nd attributes), triangular (4th attribute), and wheeled car”; soccer ball is “a ball with black dots on white background”. Such results validate that HCN can successfully learn reasonable criteria as desired. *Second*, some of the learned attributes seem to depict the same/similar concept, e.g., “spots” in the cheetah case (second row). Such results suggest that one single attribute (filter) might be insufficient for capturing the complex variations of that concept, and HCN can adaptively learn multiple attributes to better characterize such pattern diversity. *Third*, it is noteworthy that man-made objects have larger within-class variations than natural objects since they are categorized mainly based on their functionalities instead of visual appearances. Therefore, the criteria learned by HCN for man-made objects seem to be more diverse than

that for natural objects, e.g., the attributes for ambulance (fourth row) show more diversity than the attributes for cheetah and feline. Although some recently proposed methods [18], [44] are able to evaluate the degree of model interpretability, they cannot evaluate whether the model is exploiting the interpretable CNN filters in the right way, i.e., whether the model makes decisions based on the right criteria or based on biases in the dataset (e.g., co-occurring objects). Quantitatively evaluating the learned criteria inevitably requires expertise, e.g., the knowledge of visual differences between over 100 breeds of dogs in ILSVRC. We believe that such knowledge is essential for objectively evaluating all interpretable classification methods, including HCN. However, since such annotations are very expensive to label and are currently unavailable on existing datasets, we could only conduct qualitative evaluations instead. In

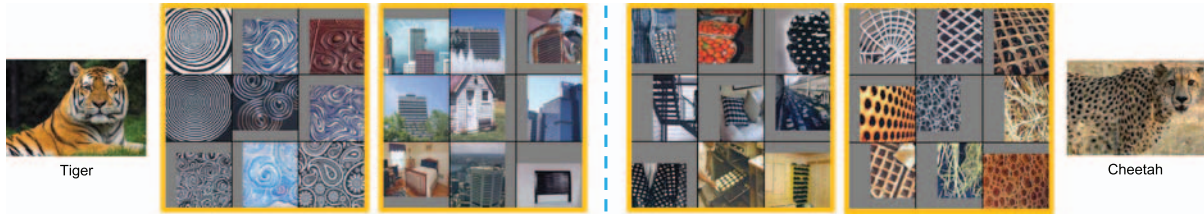


Fig. 13. Attributes that contribute differently to the two similar categories *tiger* and *cheetah* in ILSVRC. The attributes on the left help to confirm the existence of *tigers* and decreases the possibility of *cheetahs*, while the attributes on the right work oppositely.

the future, we plan to explore such expert knowledge in training and evaluating interpretable models.

Furthermore, we show that the learned criteria can give instructions to non-experts on how to distinguish similar categories. As described in Section 3, all elements in \mathbf{P} are non-negative, therefore $\mathbf{W}_{k,i} > 0$ indicates that the k th attribute contributes positively to the prediction of the i th category and vice versa. Based on this observation, we visualize some attributes that contribute differently to two very similar categories in ILSVRC (*tiger* and *cheetah*, i.e., $\mathbf{W}_{k,tiger} \cdot \mathbf{W}_{k,cheetah} < 0$), as shown in Fig. 13. The existence of attributes on the left (different kinds of stripes) increases the likelihood of *tiger* while decreases the possibility of *cheetah* (i.e., $\mathbf{W}_{k,tiger} > 0$ and $\mathbf{W}_{k,cheetah} < 0$), and the attributes on the right (different patterns of spots) work oppositely. The above result suggests that contrary to previous classification models that can only learn from manually labeled data, HCN can also give knowledge feedbacks to humans. More importantly, it is even possible that the knowledge feedbacks can help non-experts learn to distinguish fine-grained categories, which is an extremely challenging task for most ordinary people.

4.7 Feedback for Model Correction

In this part, we conduct a set of preliminary experiments on ILSVRC to evaluate the feasibility of incorporating human feedback into HCN. Based on the method described in Section 3.4, we test three schemes for selecting the attributes to be removed: 1) bottom-3k: the attributes corresponding to the 3,000 smallest values in the indicator vector \mathbf{v} (i.e., the attributes that are least useful for that category); 2) top-3k: the attributes corresponding to the 3,000 largest values (i.e., the attributes that are most useful for that category); 3) rand-3k: 3,000 randomly selected attributes. We experiment on two categories (i.e., *ambulance* and *cheetah*), and modify only one column of \mathbf{W} at a time (i.e., in each experiment, only one column of \mathbf{W} is modified). The recall (ratio of

correctly classified samples) on the corresponding categories before removing the selected classifier weights, just after removing, and after finetuning are listed in Table 8 (the overall top-1 accuracy of the three schemes on the whole validation set are given in the brackets). For both categories, we can see that: *First*, after removing the least important attributes (bottom-3k), the model performance on the corresponding category increases to a very high level yet the overall performance significantly drops, indicating that the model tends to predict samples of other categories as the selected category in that case. While on the other hand, removing other attributes (top-3k and rand-3k) only affects the performance of the selected category. *Second*, Finetuning could take the model performance back to a normal level. However, the performances of top-3k and rand-3k are worse than bottom-3k on the corresponding category, suggesting that the information carried by some of the removed attributes cannot be compensated by the remaining attributes. Such results further validate the effectiveness of HCN in disentangling the defining attributes of different categories. Note that this experiment is designed to show the potential of HCN, while other schemes (e.g., selecting by experts) are also compatible with HCN yet could be more time-consuming.




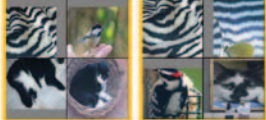

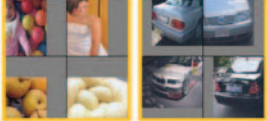



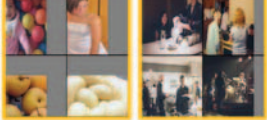
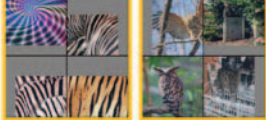
On the other hand, we also test the possibility of adding missing attributes to the existing model. The experiments are conducted on the ImageNet-150K dataset [65]. Following the experiments in Section 4.4, we additionally compare two types of input features for the classifier: (F4) the concatenated attributes of F1 and F2 (50-D). This corresponds to augmenting the learned attributes with additional manually-labeled attributes, which is designed to validate the possibility of augmenting the learned attributes with additional missing attributes; (F5) the concatenated attributes of F1-p and F2 (i.e., also 50-D). This feature also serves the same purpose as F4 but in a more realistic setting (i.e., predicting the attributes instead of manually labeling them). From the results in Table 5, we can see that: *First*, combining the learned attributes with manually-labeled attributes can improve the classification accuracy (F2 versus F4, from 59.60 to 61.55 percent), which suggests that the manually-labeled attributes are complementary with the learned attributes. Such results also validate that it is possible to improve the model performance by augmenting existing CNN models with manually-labeled attributes. *Second*, by replacing the groundtruth attribute labels with predicted attribute labels (F4 versus F5), the classification accuracy significantly drops from 61.55 to 26.35 percent. This result indicates that while the classification task could benefit from manually-labeled attributes, accurately predicting these attributes is the key for improving the accuracy.

TABLE 8
Recall on the Selected Categories After Removing the Attributes and After Finetuning

	stage	bottom-3k	top-3k	rand-3k
ambulance	beginning	76% (59.35%)	76% (59.35%)	76% (59.35%)
	removing	98% (53.17%)	0% (59.21%)	0% (59.21%)
	finetuning	84% (59.41%)	70% (59.42%)	76% (59.40%)
cheetah	beginning	88% (59.35%)	88% (59.35%)	88% (59.35%)
	removing	98% (56.94%)	0% (59.20%)	2% (59.20%)
	finetuning	88% (59.39%)	84% (59.45%)	84% (59.39%)

The overall Top-1 accuracy on the whole validation set are given in the brackets.

TABLE 9
Comparison to Previous Methods on the Criteria Learning Task

Category	Superordinate Category	USE [14]	Two-stage Alternative	HCN (ours)
	Musteline 	A musteline mammal that has stripes	A musteline mammal with 	A musteline mammal with 
		A musteline mammal that is quadrapedal, flippers, furry, ocean	A musteline mammal with 	A musteline mammal with 
	Feline 	not available	A feline with 	A feline with 

The results of [14] are directly cited from the original publication.

4.8 Comparison to Previous Methods

In this part, we compare HCN with previous methods to show the pros and cons of HCN in learning classification criteria. For this purpose, we compare to two methods on the ILSVRC dataset: 1) Unified Semantic Embedding (USE) [14]. To the best of our knowledge, USE is the only existing method that has made an attempt to learn the classification criteria. However, USE requires class-level attribute labels, which are not always available in real-world applications. Specifically, existing attribute labels on ILSVRC either contain too few (only 25) attributes for discriminating the one thousand categories [64], [65] or are too sparse and thus cannot be exploited by USE [69]. Therefore, we directly cite the results from [14] for comparison. 2) A two-stage learning alternative. This alternative is designed to simulate the situation of learning criteria on the off-the-shelf image representations learned in the classification task, i.e., the attribute learning stage and the criteria learning stage are separated in this alternative. To be specific, in the first stage, we train the upper stream of HCN with $\alpha = 1, \beta = \gamma = \delta = 0, D = 6, 144$. After that, in the second stage, the model parameters before the global max pooling layer are fixed, and the classifier weights \mathbf{W} and the whole lower stream are trained with $\alpha = 0.699, \beta = 0.299, \gamma = 0.001, \delta = 0.001$.

The results of two categories (*skunk* and *otter*) reported in [14] that overlap with ILSVRC as well as a category (*cheetah*) that is absent in [14] are shown in Table 9. Since we do not have groundtruth labels for the classification criteria, it is difficult to conduct a quantitative comparison. Therefore, we only compare the results qualitatively. For space limitation, we use the top-4 activated patches to represent the two most important attributes learned by CNN models. We can see that: *First*, compared to [14], HCN is more general, i.e., even we do not have groundtruth class-level attribute labels (the *cheetah* case), HCN can still learn reasonable criteria. On the other hand, in the case that class-level attribute labels are available (the *skunk* and *otter* case), although the attributes learned by HCN have weaker semantic meanings than the manually-defined ones,

we can still well discover the criteria. For instance, the special “black and white stripe” pattern of *skunk*, and the “water” (first attribute) and “furry” (second attribute) attributes of *otter*. *Second*, the two-stage alternative failed to learn the criteria (note that the criteria learned by the baseline method for the three categories are almost the same). This result suggests that the hierarchical structure is not encoded in such off-the-shelf “classification representations”, and thus the hierarchical classification criteria cannot be learned from these representations. Therefore, end-to-end learning or at least fine-tuning the model weights of an off-the-shelf model with HCN is quite necessary for learning the hierarchical classification criteria.

5 CONCLUSION AND FUTURE DIRECTIONS

In this paper, we propose a method named HCN to automatically discover the criteria for classification, which improves model interpretability and enables possible human feedback for model correction. With the elaborately designed network structure, HCN can learn reasonable criteria, which even have the potential of teaching people how to distinguish similar categories. Though some attributes and criteria currently learned by HCN do not entirely reflect human understanding, we still believe that this study is a promising step towards interpretable and operable CNN models. In the future, we would like to further improve the attribute interpretability by exploring the connections between human-nameable attributes and the ones learned by HCN.

In the current experiments, even with state-of-the-art model interpretation method and dataset, some interpretable attributes and classification criteria learned by CNN models are still missed, revealing the subjective nature of interpretability. Moreover, since some categories are distinguished from the others based on some non-visual or unnameable attributes, e.g., *ambulance* is different from other *cars* only partially for its visual appearance yet mainly for its utilization, and the differences between a *jaguar* and a *leopard* (Fig. 9) can hardly be

described in natural language by an ordinary person, suggesting that the interpretations produced by existing models could be hard to evaluate. However, objectively evaluating the degree of interpretability and the correctness of the interpretations are vital for interpretable methods, yet they still remain an open research problem in this field of study. In this paper, we have tried our best to design qualitative and quantitative evaluations to compare the degree of interpretability of different models, and we believe that this is a right step towards fairly comparing interpretable models. In the future, datasets with objectively and professionally labeled attributes or criteria could be a promising means for more fairly evaluating the interpretable models, yet such datasets require lots of expertise which is beyond the scope of this paper.

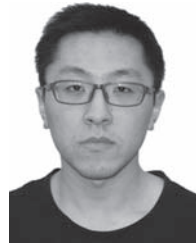
ACKNOWLEDGMENTS

This work was done at the Institute of Computing Technology, Chinese Academy of Sciences, where Haomiao Liu pursued the PhD degree. This work is partially supported by 973 Program under contract No. 2015CB351802, Natural Science Foundation of China under contracts Nos. 61390511, 61772500, Frontier Science Key Research Project CAS No. QYZDJ-SSW-JSC009, and Youth Innovation Promotion Association CAS No. 2015085.

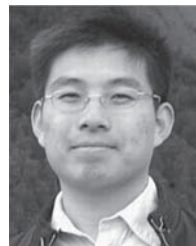
REFERENCES

- [1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. 25th Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
- [2] C. Szegedy et al., "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1–9.
- [3] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets," in *Proc. 18th Int. Conf. Artif. Intell. Statist.*, 2015, pp. 562–570.
- [4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Representations*, 2015, pp. 1–14.
- [5] 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. 770–778.
- [6] J. Wang, Z. Wei, T. Zhang, and W. Zeng, "Deeply-fused nets," 2016, *arXiv:1605.07716*, pp. 1–16.
- [7] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2261–2269.
- [8] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7132–7141.
- [9] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8697–8710.
- [10] J. Gu et al., "Recent advances in convolutional neural networks," *Pattern Recognit.*, vol. 77, pp. 354–377, 2018.
- [11] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: A simple and accurate method to fool deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2574–2582.
- [12] L. Chang and D. Y. Tsao, "The code for facial identity in the primate brain," *Cell*, vol. 169, no. 6, pp. 1013–1028, 2017.
- [13] E. Rosch, "Principles of categorization," in *Concepts: Core Readings*, E. Margolis and S. Laurence, Eds. Cambridge, MA, USA: MIT Press, 1999, pp. 189–206.
- [14] S. J. Hwang and L. Sigal, "A unified semantic embedding: Relating taxonomies and attributes," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 271–279.
- [15] B. Zhou, Y. Sun, D. Bau, and A. Torralba, "Interpretable basis decomposition for visual explanation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 122–138.
- [16] R. R. Selvaraju et al., "Choose your neuron: Incorporating domain knowledge through neuron-importance," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 540–556.
- [17] Q. Zhang, Y. N. Wu, and S.-C. Zhu, "Interpretable convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8827–8836.
- [18] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, "Network dissection: Quantifying interpretability of deep visual representations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 3319–3327.
- [19] R. C. Fong and A. Vedaldi, "Interpretable explanations of black boxes by meaningful perturbation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 3429–3437.
- [20] C. Huang, C. C. Loy, and X. Tang, "Unsupervised learning of discriminative attributes and visual representations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 5175–5184.
- [21] V. Escorcia, J. C. Niebles, and B. Ghanem, "On the relationship between visual attributes and convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1256–1264.
- [22] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth, "Describing objects by their attributes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 1778–1785.
- [23] A. Farhadi, I. Endres, and D. Hoiem, "Attribute-centric recognition for cross-category generalization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2010, pp. 2352–2359.
- [24] C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 951–958.
- [25] K. Duan, D. Parikh, D. Crandall, and K. Grauman, "Discovering localized attributes for fine-grained recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3474–3481.
- [26] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid, "Label-embedding for attribute-based classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 819–826.
- [27] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid, "Label-embedding for image classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 7, pp. 1425–1438, Jul. 2016.
- [28] S. Vittayakorn, T. Umeda, K. Murasaki, K. Sudo, T. Okatani, and K. Yamaguchi, "Automatic attribute discovery with neural activations," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 252–268.
- [29] G. Griffin and P. Perona, "Learning and using taxonomies for fast visual categorization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2008, pp. 1–8.
- [30] M. Marszałek and C. Schmid, "Constructing category hierarchies for visual recognition," in *Proc. Eur. Conf. Comput. Vis.*, 2008, pp. 479–491.
- [31] J. Deng, J. Krause, A. C. Berg, and L. Fei-Fei, "Hedging your bets: Optimizing accuracy-specificity trade-offs in large scale visual recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3450–3457.
- [32] V. Ordonez, J. Deng, Y. Choi, A. C. Berg, and T. L. Berg, "From large scale image categorization to entry-level categories," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 2768–2775.
- [33] J. Deng et al., "Large-scale object classification using label relation graphs," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 48–64.
- [34] N. Ding, J. Deng, K. P. Murphy, and H. Neven, "Probabilistic label relation graphs with ising models," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1161–1169.
- [35] B. Zhao, F. Li, and E. P. Xing, "Large-scale category structure aware image categorization," in *Proc. 24th Int. Conf. Neural Inf. Process. Syst.*, 2011, pp. 1251–1259.
- [36] N. Srivastava and R. R. Salakhutdinov, "Discriminative transfer learning with tree-based priors," in *Proc. 26th Int. Conf. Neural Inf. Process. Syst.*, 2013, pp. 2094–2102.
- [37] Z. Yan, H. Zhang, R. Piramuthu, V. Jagadeesh, D. DeCoste, W. Di, and Y. Yu, "HD-CNN: Hierarchical deep convolutional neural networks for large scale visual recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 2740–2748.
- [38] W. Goo, J. Kim, G. Kim, and S. J. Hwang, "Taxonomy-regularized semantic deep convolutional neural networks," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 86–101.
- [39] K. Ahmed, M. H. Baig, and L. Torresani, "Network of experts for large-scale image categorization," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 516–532.
- [40] O. Russakovsky et al., "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vision*, vol. 115, no. 3, pp. 211–252, 2015.
- [41] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 818–833.
- [42] A. Dosovitskiy and T. Brox, "Inverting visual representations with convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 4829–4837.

- [43] A. Dosovitskiy and T. Brox, "Generating images with perceptual similarity metrics based on deep networks," in *Proc. 30th Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 658–666.
- [44] R. Fong and A. Vedaldi, "Net2Vec: Quantifying and explaining how concepts are encoded by filters in deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8730–8738.
- [45] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," 2013, *arXiv:1312.6034*, pp. 1–8.
- [46] S. Lapuschkin, A. Binder, G. Montavon, K.-R. Müller, and W. Samek, "Analyzing classifiers: Fisher vectors and deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2912–2920.
- [47] L. M. Zintgraf, T. S. Cohen, T. Adel, and M. Welling, "Visualizing deep neural network decisions: Prediction difference analysis," in *Proc. Int. Conf. Learn. Representations*, 2017, pp. 1–12.
- [48] O. Li, H. Liu, C. Chen, and C. Rudin, "Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions," 2017, *arXiv:1710.04806*, pp. 1–8.
- [49] D. Mascharka, P. Tran, R. Soklaski, and A. Majumdar, "Transparency by design: Closing the gap between performance and interpretability in visual reasoning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4942–4950.
- [50] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proc. Int. Conf. Mach. Learn.*, 2010, pp. 807–814.
- [51] M. Lin, Q. Chen, and S. Yan, "Network in network," in *Proc. Int. Conf. Learn. Representations*, 2014, pp. 1–10.
- [52] A. Krizhevsky, "Learning multiple layers of features from tiny images," 2009. [Online]. Available: <https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf>
- [53] Y. Jia et al., "Caffe: Convolutional architecture for fast feature embedding," in *Proc. ACM Int. Conf. Multimedia*, 2014, pp. 675–678.
- [54] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.
- [55] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.
- [56] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. Int. Conf. Mach. Learn. Workshop*, on Deep Learning for Audio, Speech and Language Processing, 2013, pp. 1–6.
- [57] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The caltech-UCSD birds-200–2011 dataset," California Institute of Technology, Tech. Rep. CNS-TR-2011-001, 2011.
- [58] G. Patterson and J. Hays, "COCO attributes: Attributes for people, animals, and objects," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 85–100.
- [59] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 3730–3738.
- [60] G. Patterson, C. Xu, H. Su, and J. Hays, "The sun attribute database: Beyond categories for deeper scene understanding," *Int. J. Comput. Vis.*, vol. 108, no. 1/2, pp. 59–81, 2014.
- [61] R. Ranjan, C. D. Castillo, and R. Chellappa, "L2-constrained softmax loss for discriminative face verification," 2017, *arXiv:1703.09507*, pp. 1–10.
- [62] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, "Understanding neural networks through deep visualization," in *Proc. Int. Conf. Mach. Learn. Workshop*, on Deep Learning, 2015, pp. 1–12.
- [63] M. Rastegari, A. Farhadi, and D. Forsyth, "Attribute discovery via predictable discriminative binary codes," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 876–889.
- [64] O. Russakovsky and L. Fei-Fei, "Attribute learning in large-scale datasets," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 1–14.
- [65] H. Liu, R. Wang, S. Shan, and X. Chen, "Learning multifunctional binary codes for both category and attribute oriented retrieval tasks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 3901–3910.
- [66] W. Liu, Y. Wen, Z. Yu, and M. Yang, "Large-margin softmax loss for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 507–516.
- [67] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. Nov, pp. 2579–2605, 2008.
- [68] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 499–515.
- [69] W. Ouyang, H. Li, X. Zeng, and X. Wang, "Learning deep representation with large-scale attributes," in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 1895–1903.



Haomiao Liu received the BS degree in electrical engineering from the Beijing University of Technology, Beijing, China, in 2013 and the PhD degree in computer science from the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS). He was associated with the Key Lab of Intelligent Information Processing, Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), Beijing, China, and the University of Chinese Academy of Sciences (UCAS), Beijing, China. He is a member of the IEEE.



Ruiping Wang received the BS degree in applied mathematics from Beijing Jiaotong University, Beijing, China, in 2003 and the PhD degree in computer science from the Institute of Computing Technology (ICT), Chinese Academy of Science (CAS), Beijing, in 2010. He was a postdoctoral researcher with the Department of Automation, Tsinghua University, Beijing, from July 2010 to June 2012. He also spent one year working as a research associate with the Computer Vision Laboratory, Institute for Advanced Computer Studies, University of Maryland, College Park, from November 2010 to October 2011. He has been with the faculty of the Institute of Computing Technology, Chinese Academy of Sciences, since July 2012, where he is currently a professor. His interests include computer vision, pattern recognition, and machine learning. He is a member of the IEEE.



Shiguang Shan is a professor of ICT, CAS, and the deputy director with the Key Laboratory of Intelligent Information Processing, CAS. His research interests cover computer vision, pattern recognition, and machine learning. He has authored more than 200 papers in refereed journals and proceedings in the areas of computer vision and pattern recognition. He was a recipient of the Chinas State Natural Science Award, in 2015, and the Chinas State S&T Progress Award, in 2005 for his research work. He has served as the area chair for many international conferences, including ICCV11, ICPR12, ACCV12, FG13, ICPR14, and ACCV16. He is an associate editor of several journals, including the *IEEE Transactions on Image Processing*, *Computer Vision and Image Understanding*, *Neurocomputing*, and the *Pattern Recognition Letters*. He is a senior member of the IEEE.



Xilin Chen is a professor with the Institute of Computing Technology, Chinese Academy of Sciences (CAS). He has authored one book and more than 300 papers in refereed journals and proceedings in the areas of computer vision, pattern recognition, image processing, and multimodal interfaces. He is currently an associate editor of the *IEEE Transactions on Multimedia*, and a senior editor of the *Journal of Visual Communication and Image Representation*, a leading editor of the *Journal of Computer Science and Technology*, and an associate editor-in-chief of the *Chinese Journal of Computers*, and the *Chinese Journal of Pattern Recognition and Artificial Intelligence*. He served as an organizing committee member for many conferences, including general co-chair of FG13 / FG18, program co-chair of ICMI 2010. He is / was an area chair of CVPR 2017 / 2019 / 2020, and ICCV 2019. He is a fellow of the ACM, IEEE, IAPR, and CCF.

▷ For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/csdl.