Interpreting CNNs via Decision Trees

Quanshi Zhang†, Yu Yang‡, Haotian Ma§, and Ying Nian Wu‡
†Shanghai Jiao Tong University, ‡University of California, Los Angeles, §South China University of Technology

Abstract

This paper aims to quantitatively explain the rationales of each prediction that is made by a pre-trained convolutional neural network (CNN). We propose to learn a decision tree, which clarifies the specific reason for each prediction made by the CNN at the semantic level. I.e. the decision tree decomposes feature representations in high conv-layers of the CNN into elementary concepts of object parts. In this way, the decision tree tells people which object parts activate which filters for the prediction and how much each object part contributes to the prediction score. Such semantic and quantitative explanations for CNN predictions have specific values beyond the traditional pixel-level analysis of CNNs. More specifically, our method mines all potential decision modes of the CNN, where each mode represents a typical case of how the CNN uses object parts for prediction. The decision tree organizes all potential decision modes in a coarse-to-fine manner to explain CNN predictions at different fine-grained levels. Experiments have demonstrated the effectiveness of the proposed method.

1. Introduction

Convolutional neural networks (CNNs) [20, 18, 14] have achieved superior performance in various tasks. However, besides the discrimination power, model interpretability is still a significant challenge for neural networks. Many studies have been proposed to visualize or analyze feature representations hidden inside a CNN, in order to open the black box of neural networks.

Motivation & objective: In the scope of network interpretability, state-of-the-art algorithms are still far from the ultimate goal of explaining why a CNN learns knowledge as it is. Although some theories, such as the information bottleneck [34], analyzed statistical characteristics of a neural network, it is still a challenge to explain why a CNN encodes frontal-leg features, rather than rear-leg features, for classification during the end-to-end learning process.

Therefore, in this study, we limit our discussion to the issue of explaining what knowledge a CNN learns. In this direction, our research focuses on the following two new perspectives of interpreting CNNs:

• How to explain features of a middle layer in a CNN at the semantic level. I.e. we aim to transform chaotic features of filters inside a CNN into semantically meaningful concepts, such as object parts, so as to help people to understand the knowledge in the CNN.
• How to quantitatively analyze the rationale of each CNN prediction. We need to figure out which filters/parts pass their information through the CNN and contribute to the prediction output. We also report the numerical contribution of each filter (or object part) to the output score.

As shown in Fig. 1, the above two perspectives are crucial in real applications and have essential differences from traditional pixel-level visualization and diagnosis of CNN features [37, 23, 10, 24, 22]. Our semantic and quantitative...
explanations for CNNs have potential values beyond pixel-level visualization/analysis of CNNs. Semantic and quantitative explanations can help people better understand and trust CNN predictions. E.g. in critical applications, such as the recommendation for a surgery plan, people are usually not simply satisfied by the plan itself, but expect a quantitative explanation for the plan.

However, bridging the gap between a CNN’s middle-layer features and semantic explanations has not been well explored yet. We will introduce our task, clarify its challenges, and define relevant concepts from the following two perspectives.

- **Bridging middle-layer features with semantic concepts:** Given an input image, the first issue is to learn more interpretable feature representations inside a CNN and associate each neural activation inside a CNN with a semantic concept. This presents significant challenges for state-of-the-art algorithms.

Firstly, we need to force feature representations in middle conv-layers to be well disentangled during the learning process. According to \cite{2, 38}, a filter in traditional CNNs usually represents a mixture of parts and textures. Learning semantically meaningful filters is difficult but is the foundation of semantic-level explanations. In this research, we learn a CNN with disentangled filters in high conv-layers. Each filter needs to be consistently activated by the same object region over different input images. We do not use any annotations of parts or textures to supervise the disentanglement of filter features.

Secondly, we also need to associate each disentangled filter with an explicit semantic meaning (i.e. an object part in this study). This association enables linguistic descriptions of middle-layer knowledge, for example, how many parts are memorized in the CNN and how the parts are organized.

- **Bridging middle-layer features with final CNN predictions:** When we have assigned middle-layer features with specific part concepts, the next issue is to quantitatively explain how the CNN uses these middle-layer features to compute prediction scores. In other words, given an input image, we hope to clarify the specific rationale of the CNN prediction.

Here, we define the **rationale** of a CNN prediction as the set of object parts (or filters) that are activated and contribute to the prediction. Given different input images, the CNN uses different object parts to activate different sets of filters to compute prediction scores, thereby having different rationales. Let us take the bird classification as an example. The CNN may use several filters activated by the head appearances as rationales to classify a standing bird, and the CNN may take filters for wings to distinguish a flying bird.

Given each input image, our task is to clarify filters of which object parts are activated and to quantitatively measure the contribution of each object part to the prediction. The concept of the contribution was also called the “importance” in \cite{24, 22} and was termed “attribution” in \cite{16}. As shown in Fig. 2, we describe the contribution as “a head filter contributes 2.32%, and a feet filter contributes 0.72%.”

**Task:** As shown in Fig. 1, given a pre-trained CNN, we propose a method to construct a decision tree to explain CNN predictions semantically and quantitatively. We summarize rationales of CNN predictions on all images into various **decision modes**. Each tree node represents a decision mode. Each decision mode describes common rationales of predictions that are shared by multiple images. I.e. for these images, the CNN usually activates similar filters (object parts), and each part makes a similar contribution to the prediction.

The decision tree hierarchically represents all potential decision modes of a CNN in a coarse-to-fine manner. Nodes near to the tree root node mainly represent most common decision modes (prediction rationales) shared by many images. Nodes near leaves correspond to fine-grained modes of minority images. In particular, each leaf node encodes the specific decision mode of a certain image.

In order to build the decision tree, we learn filters to represent object parts (we do not label any parts or textures as additional supervision). Then, we assign each filter with a specific part name. Finally, we mine decision modes to

\footnote{Zhang et al. \cite{40} summarized the six types of semantics defined in \cite{2} as parts and textures.}
explain how the CNN use parts/filters for prediction, and construct a decision tree.

**Inference:** When the CNN makes a prediction for an input image, the decision tree determines a parse tree (see green lines in Fig. 2) to encode a series of explanations. Each node (decision mode) in the parse tree quantitatively explains the prediction at a different abstraction level, i.e., clarifying how much each object part/filter contributes to the prediction score.

**Compared to fine-grained modes in leave nodes, we are more interested in generic decision modes in high-level nodes.** Generic decision modes usually select significant object parts (filters) as the rationale of CNN predictions and ignore insignificant ones. Thus, generic decision modes reflect compact rationales for CNN predictions.

**Contributions:** In this paper, we aim to use semantic visual concepts to explain CNN predictions quantitatively and semantically. We propose to learn the decision tree without strong supervision for explanations. Our method is a generic approach and has been successfully applied to various benchmark CNNs. Experiments have demonstrated the effectiveness of our method.

2. Related work

In this section, we limit our discussion to the literature of opening the black box of CNN representations. [2, 21, 9, 4] discussed the definition of interpretability from different perspectives concerning different tasks. Zhang et al. [43] made a survey for the interpretability of deep visual models.

**CNN visualization:** Visualization of filters in a CNN is the most direct way of exploring the pattern hidden inside a neural unit. Gradient-based visualization [37, 23] estimates the input image that maximizes the activation score of a neural unit. Up-convolutional nets [10] invert feature maps of conv-layers into images. Unlike gradient-based methods, up-convolutional nets cannot mathematically ensure that the visualization result reflects actual neural representations.

Zhou et al. [44] proposed a method to accurately compute the image-resolution receptive field of neural activations in a feature map. The estimated receptive field of a neural activation is smaller than the theoretical receptive field based on the filter size. The accurate estimation of the receptive field is crucial to understand a filter’s representations. Bau et al. [2] further defined six types of semantics for CNNs, i.e., objects, parts, scenes, textures, materials, and colors. Zhang et al. [38] summarized the six types of semantics into “parts” and “textures.” Nevertheless, each filter in a CNN represents a mixture of semantics. [45] explained semantic reasons for visual recognition.

**Network diagnosis:** Beyond visualization, some methods diagnose a pre-trained CNN to obtain insight understanding of CNN representations.

Fong and Vedaldi [11] analyzed how multiple filters jointly represented a specific semantic concept. Yosinski et al. [36] evaluated the transferability of filters in intermediate conv-layers. Aubry et al. [1] computed feature distributions of different categories in the CNN feature space. Selvaraju et al. [26] and Fong et al. [12] propagated gradients of feature maps w.r.t. the CNN loss back to the image, in order to estimate image regions that directly contribute the network output. The LIME [24] and SHAP [22] extracted image regions that were used by a CNN to predict a label. Zhang et al. [41] used an explainer network to interpret object-part representations in intermediate layers of CNNs.

Network-attack methods [17, 30] diagnosed network representations by computing adversarial samples for a CNN. In particular, influence functions [17] were proposed to compute adversarial samples, provide plausible ways to create training samples to attack the learning of CNNs, fix the training set, and further debug representations of a CNN. Lakkaraju et al. [19] discovered knowledge blind spots (unknown patterns) of a pre-trained CNN in a weakly-supervised manner. The study of [39] examined representations of conv-layers and automatically discover potential, biased representations of a CNN due to the dataset bias.


**Decision trees for neural networks:** Distilling knowledge from neural networks into tree structures is an emerging direction [13, 31, 5], but the trees did not explain the network knowledge at a human-interpretable semantic level. Wu et al. [35] learned a decision tree via knowledge distillation to represent the output feature space of an RNN, in order to regularize the RNN for better representations. Vaughan et al. [32] distilled knowledge into an additive model for explanation.

Despite the use of tree structures, there are two main differences between the above two studies and our research. Firstly, we focus on using a tree to explain each prediction made by a pre-trained CNN semantically. In contrast, decision trees in the above studies are mainly learned for clas-
3. Image-specific rationale of a CNN prediction

In this section, we design a method to simplify the complex feature processing inside a CNN into a linear form (i.e., Equation (3)), as the specific rationale of the prediction w.r.t. the input image. This linear equation clarifies (i) which object parts activate which filters in the CNN and (ii) how much these parts/filters contribute to the final prediction score.

In order to obtain semantic-level rationale of a CNN prediction, we need (i) first to ensure that the CNN’s middle-layer features are semantically meaningful, and (ii) then to extract explicit contributions of middle-layer features to the prediction score.

In this study, we learn the CNN for object classification. Theoretically, we can interpret CNNs oriented to different tasks. Nevertheless, in this paper, we limit our attention to CNNs for classification, in order to simplify the story.

3.1. Learning disentangled filters

The basic idea is to revise a benchmark CNN, in order to make each filter in the top conv-layer represent a specific object part. We expect the filter to be automatically converged to the representation of a part, instead of using additional part annotations to supervise the learning process.

We apply the filter loss [40] to each filter in the top conv-layer to push the filter towards the representation of an object part. As shown in Fig. 3, the filter is learned to be activated by the same object part given different input images. Theoretically, our method also supports other techniques of mining interpretable features in middle layers [38, 27]. Nevertheless, the filter loss usually ensures more meaningful features than the other approaches.

**Filter loss:** Let \( x_f \in \mathbb{R}^{h \times w \times d} \) denote the feature map of a filter \( f \). Without part annotations, the filter loss forces \( x_f \) to be exclusively activated by a specific part of a category. We can summarize the filter loss as the minus mutual information between the distribution of feature maps and that of part locations.

\[
\text{Loss}_f = \sum_{x_f \in X_f} \text{Loss}(x_f) = -MI(X_f; P) = - \sum_{\mu \in P} p(\mu) \sum_{x_f \in X_f} p(x_f | \mu) \log \frac{p(x_f | \mu)}{p(x_f)} \tag{1}
\]

where \( MI(\cdot) \) indicates the mutual information. \( X_f \) denotes a set of feature maps of \( f \) extracted from different input images. \( P = \{ \mu | \mu = [h, w], 1 \leq h, w \leq L \} \cup \{ \emptyset \} \) is referred to as a set of all part-location candidates. Each location \( \mu = [h, w] \) corresponds to an activation unit in \( x_f \). Besides, \( \emptyset \in P \) denotes the case that the target part does not appear in the input image. In this case, all units in \( x_f \) are expected to keep inactivated. The joint probability \( p(x_f, \mu) \) to describe the compatibility between \( x_f \) and \( \mu \) (please see [40] for details).

The filter loss ensures that given an input image, \( x_f \) should match only one of all \( L^2 + 1 \) location candidates. It is assumed that repetitive shapes on various regions are more likely to describe low-level textures than high-level parts. If the part appears, \( x_f \) should have a single activation peak at the part location; otherwise, \( x_f \) should keep inactivated.

3.2. Quantitative rationales of CNN predictions

As analyzed in [2], filters in high conv-layers are more prone to represent object parts, while those in low conv-layers usually describe textures. Therefore, we choose filters in the top conv-layer to represent object parts. Consequently, we quantitatively analyze how fully-connected (FC) layers use object-part features from the top conv-layer to make final predictions, as the rationale.

Given an input image \( I \), let \( x \in \mathbb{R}^{L \times L \times D} \) denote the feature map of the top conv-layer after a ReLU operation, where \( L \) denotes the scale of the feature map, and \( D \) is the filter number. Let \( y \) denote the scalar classification score of a certain category before the softmax operation (when the CNN is learned for multiple categories, we may learn a specific decision tree to explain the output of each category). Our task is to use \( x \) to represent the rationale of \( y \).

As discussed in [22, 24], we can use a piecewise linear representation to represent the function of cascaded FC layers and ReLU layers, as follows.

\[
y = f_{\text{fc-o}}(f_{\text{fc-i}}(\cdots f_{\text{fc-1}}(x)))) = \sum_{h, w, d} g^{(h, w, d)} \cdot x^{(h, w, d)} + b \tag{2}
\]

where \( x^{(h, w, d)} \in \mathbb{R} \) denotes the element at the location \((h, w)\) of the \(d\)-th channel; \( g^{(h, w, d)} \) is a weight that describes the importance of \( x^{(h, w, d)} \) for the prediction on \( I \). Theoretically, we can compute \( g = \frac{\partial y}{\partial x} \) and \( b = y - g \odot x \).

We use weights \( g \) to denote the specific rationale of the prediction for the input image. \( g^{(h, w, d)} \cdot x^{(h, w, d)} \) measures \( x^{(h, w, d)} \)'s quantitative contribution to the prediction.

Different input images correspond to different weights \( g \), i.e. different rationales of their CNN predictions. It is
4. Learning a decision tree

We learn a decision tree to interpret the classification of each category. In the following two subsections, we first define basic concepts in a decision tree and then introduce the learning algorithm.

4.1. Decision tree

Let us focus on the decision tree for a certain category. We consider images of this category as positive images and consider other images as negative images. \( \Omega^+ \) denotes image indexes of the target category, i.e., positive images, and \( \Omega = \Omega^+ \cup \Omega^- \) represents all training images. For an image \( I_i, i \in \Omega \), \( y_i \) denotes the classification score of the target category before the softmax layer.

As shown in Fig. 2, each node \( v \) in the decision tree encodes a decision mode that is hidden inside FC layers of the CNN. A decision mode represents a common rationale of the prediction shared by a group of positive training images \( \Omega_v \subset \Omega^+ \). The decision tree organizes the hierarchy of decision modes in a coarse-to-fine manner from the root node to leaf nodes. Children nodes \( v' \in Child(v) \) divides the parent \( v \)'s decision mode into fine-grained modes. Fine-grained modes are shared by sub-groups of images.

Just like the rationale defined in Equation (3), the decision mode in node \( v \) is parameterized with \( w \) and \( b \), and the mode explains predictions on a certain set of images \( \Omega_v \). For each image \( I_i, i \in \Omega_v \), the decision mode is given as:

\[
b_v(x_i) = w^T x_i + b \quad w = \alpha \odot \mathcal{g} \quad \text{(4)}
\]

\[
\max_{\mathcal{g}} \sum_{i \in \Omega_v} \text{cosine}(\mathcal{g}, \mathcal{g}). \quad \text{s.t.} \quad \mathcal{g}^T \mathcal{g} = 1 \quad \text{(5)}
\]

\[
\min_{\alpha, b} \frac{1}{|\Omega_v|} \sum_{i \in \Omega_v} (w^T x_i + b - y_i)^2 + \lambda \left\| \alpha \right\|_1 \quad \text{(6)}
\]

where \( \mathcal{g} \) is referred to as the rationale of the decision mode. \( \mathcal{g} \) is a unit vector (||\( g ||_2 = 1 \)) that reflects common rationales that are shared by all images in \( \Omega_v \). \( \alpha \in \{0, 1\}^D \) is given as a binary selection of filters in the decision mode. \( \odot \) denote element-wise multiplications. We compute sparse \( \alpha \) to obtain sparse explanations for the decision mode.\(^4\)

In particular, when \( v \) is a leaf node, the decision mode is formulated as the rationale of a specific image \( I_i \). I.e. \( \alpha = \{1, \ldots, 1\}^T \) and \( w = \alpha \odot g_i = g_i \), where \( g_i \) is computed in Equation (3).

4.2. Learning decision trees

Just like hierarchical clustering, the basic idea of learning a decision tree is to summarize common generic decision modes from specific decision modes of different images. Algorithm 1 shows the pseudo-code of the learning process. At the beginning, we initialize the decision mode \( g_i \) of each positive image \( I_i \) as a leaf node by setting \( \mathcal{g} = g_i \), and \( \alpha = 1 \). Thus, we build an initial tree \( Q \) as shown in Fig. 4, in which the root node takes decision modes of all positive images as children. Then, in each step, we select and merge two nodes \( v, v' \in V \) in the second tree layer (i.e., children of the root node) to obtain a new node \( u \), where \( V \) denotes the children set of the root. \( u \) becomes a new child
of the root node, and \( v \) and \( v' \) are re-assigned as 'u’s children. The image set of \( u \) is defined as \( \Omega_u = \Omega_v \cup \Omega_{e'} \) and we learn \( \alpha, \beta, \gamma \) for \( u \) based on Equations (5) and (6).

In this way, we gradually revise the initial tree \( P_0 = Q \) towards the final tree after \( T \) merging operations as

\[
Q = P_0 \rightarrow P_1 \rightarrow P_2 \rightarrow \cdots \rightarrow P_T = \hat{P}
\]

(7)

We formulate the objective for learning as follows.

\[
\max_{\hat{P}_i} E = \frac{\prod_i p(x_i)}{\prod_i q(x_i)} e^{-\beta\|x_i\|} \quad \text{Diversity of decision nodes}
\]

(8)

where \( p(x_i) \) denotes the likelihood of \( x_i \) being positive that is estimated by the tree \( P \). \( \beta \) is a scaling parameter\(^5\).

This objective penalizes the decrease of the discriminative power and forces the system to summarize a few generic decision modes for explanation. We compute the likelihood of \( x_i \) being positive as

\[
P(x_i) = e^{\bar{h}(x_i)} / \sum_{j \in \Omega} e^{\bar{h}(x_j)}
\]

(9)

where \( \bar{h}(x_i) = h_i(x_i) \) denotes the prediction on \( x_i \) based on best child \( \hat{v} \in V \) in the second tree layer. \( \gamma \) is a constant scaling parameter\(^5\).

In the \( t \)-th step, we merge two nodes \( v, v' \in V \) in the second tree layer of \( P_{t-1} \) to get a new node \( u \), thereby obtaining a new tree \( P_t \). We can easily compute \( \Delta \log E \) w.r.t. each pair of \( (v, v') \) based on Equation (8). Thus, we learn the decision tree via a greedy strategy. In each step, we select and merge the nodes \( v, v' \in V \) that maximize \( \frac{\Delta \log E}{\|x_{\hat{v}}\|\|x_v\|\|x_{v'}\|} \). We normalize \( \Delta \log E \) for reasonable clustering performance. Because each node merger operation only affects \( \bar{h}(x_i) \) values of a few examples in \( \Omega_v \cup \Omega_{e'} \), we can quickly estimate \( \Delta \log E \) for each pair of nodes \( (v, v') \).

### 4.3. Interpreting CNNs

Given a testing image \( I_i \), the CNN makes a prediction \( y_i \). The decision tree estimates quantitative decision modes of the prediction at different fine-grained levels. During the inference procedure, we can infer a parse tree, which starts from the root node, in a top-down manner. Green lines in Fig. 4 show a parse tree. When we select the decision mode in the node \( v \) as the rationale, we can further select its child \( \hat{v} \) that maximizes the compatibility with the most specific rationale \( g_i \) as a more fine-grained mode:

\[
\hat{v} = \arg\max_{v \in \text{Child}(u)} \cosine(g_i, w_v)
\]

(10)

where we add the subscript \( v \) to differentiate the parameter of \( v \) from parameters of other nodes.

A node \( v \) in the parse tree provides the rationale of the prediction on image \( I_i \) at a certain fine-grained level. We compute the vector \( \rho_i \) and \( \hat{\rho}_i \) to evaluate the contribution of different filters and that of different object parts.

\[
\rho_i = w_v \circ x_i, \quad \hat{\rho}_i = A \rho_i
\]

(11)

where the \( d \)-th element of \( \rho_i \in \mathbb{R}^D, \rho_i^{(d)} \), denotes the contribution to the CNN prediction that is made by the \( d \)-th filter. If \( \rho_i^{(d)} > 0 \), then the \( d \)-th object part makes a positive contribution. If \( \rho_i^{(d)} < 0 \), then the \( d \)-th filter makes a negative contribution. Based on visualization results in Figs. 3 and 6, we label a matrix \( A \in \{0, 1\}^{M \times D} \) to assign each filter in the top conv-layer with a specific object part, where \( M \) is the part number. Each filter is assigned to a certain part, and the annotation cost is \( O(M) \). Similarly, the \( m \)-th element of \( \hat{\rho}_i \in \mathbb{R}^M, \hat{\rho}_i^{(m)} \) measures the contribution of the \( m \)-th part.

### 5. Experiments

#### Implementation details:
We learned four types of disentangled CNNs based on structures of four benchmark CNNs, including the AlexNet [18], the VGG-M network [29], the VGG-S network [29], the VGG-16 network [29]. Note that as discussed in [40], the filter loss in the explainer is not compatible with skip connections in residual networks [14]. We followed the technique of [40] to modify an ordinary CNN to a disentangled CNN, which changed the top conv-layer of the CNN to a disentangled conv-layer and further added a disentangled conv-layer on the top conv-layer. We used feature maps of the new top conv-layer as the input of our decision tree. We loaded parameters of all old conv-layers directly from the CNN that was pre-trained using images in the ImageNet ILSVRC 2012 dataset [8] with a loss for 1000-category classification. We initialized parameters of the new top conv-layer and all FC layers. Inspired by previous studies of [40], we can weaken the problem of multi-category classification as a number of single-category classifications to simplify the evaluation of interpretability. Thus, we fine-tuned the CNN for binary classification of a single category from random images with the log logistic loss using three benchmark datasets. We simply set parameters as \( \beta = 1, \gamma = 1 / \mathbb{E}_{i \in \Omega} \|g_i\|, \) and \( \lambda = 10^{-6} \sqrt{\|f_i\|} \) in all experiments for fair comparisons.

#### Datasets:
Because the quantitative explanation of CNN predictions requires us to assign each filter in the top conv-layer with a specific object part, we used three benchmark datasets with ground-truth art annotations to evaluate our method. The selected datasets include the PASCAL-Part Dataset [7], the CUB200-2011 dataset [33], and the ILSVRC-2013 DET Animal-Part dataset [38]. Just like in most part-localization studies [7, 38], we used animal categories, which prevalently contain non-rigid shape deformation, for evaluation. I.e. we selected six animal categories—bird, cat, cow, dog, horse, and sheep—from the PASCAL Part Dataset. The CUB200-2011 dataset contains 11.8K images of 200 bird species. Like in [3, 28], we ignored species labels and regarded all these images as a single bird category. The ILSVRC 2013 DET Animal-Part dataset [38] consists of 30 animal categories among all the 200 categories for object detection in the ILSVRC 2013 DET dataset [8].

5 Please see the experiment section for settings of \( \beta, \gamma, \) and \( \lambda \).
The vector $\hat{\varphi}_i$ in Equation (11) specifies the contribution of different object parts in the prediction of $y_i$. For the $m$-th object part, we computed $\text{contri}_m = |\hat{\varphi}_i^{(m)}| / \sum_{m'=1}^M |\hat{\varphi}_i^{(m')}|$ as the ratio of the part’s contribution.

More specifically, for CNNs based on the ILSVRC 2013 DET Animal-Part dataset, we manually labeled the object part for each filter in the top conv-layer. For CNNs based on the Pascal VOC Part dataset [7], the study of [40] merged tens of small parts into several major landmark parts for the six animal categories. Given a CNN for a certain category, we used [44] to estimate regions in different images that correspond to each filter’s neural activations, namely the image receptive field of the filter (please see Figs. 6 and 3). For each filter, we selected a part from all major landmark parts, which was closest to the filter’s image receptive field through all positive images. For the CNN based on the CUB200-2011 dataset, we used ground-truth positions of the breast, forehead, nape, tail of birds as major landmark parts. Similarly, we assigned each filter in the top conv-layer with the nearest landmark part.

**Evaluation metrics:** The evaluation has two aspects. Firstly, we use two metrics to evaluate the accuracy of the estimated rationale of a prediction. The first metric evaluates errors of object-part contributions to the CNN prediction that were estimated using nodes in the second tree layer. Given an input image $I$, $\hat{\varphi}_i$ in Equation (11) denotes the quantitative contribution of the $i$-th part. Accordingly, $\hat{\varphi}_i = y - \hat{y}_i$ is referred to as the ground-truth contribution of the part, where $y$ denotes the original CNN prediction on $I$; $\hat{y}_i$ is the output when we removed neural activations from feature maps (filters) corresponding to the $i$-th part. In this way, we used the deviation $\|\hat{\varphi}_i - \hat{\varphi}_i|I_{I=1}[y] \|$ to denote the error of the $i$-th part contributions. Another metric, namely the fitness of contribution distributions, compares the ground-truth contribution distribution over different filters in the top conv-layer with the estimated contribution of these filters during the prediction process. When the decision tree uses node $\hat{c}$ to explain the prediction for $I_i$, the vector $\rho_i$ in Equation (11) denotes the estimated contribution distribution of different filters. $\mathbf{t}_i = \mathbf{g} \circ \mathbf{x}_i$ corresponds to the ground-truth contribution distribution. We reported the interaction-of-the-union value between $\rho_i$ and $\mathbf{t}_i$ to measure the fitness of the ground-truth and the estimated filter contribution distributions. *I.e.* we computed the fitness as $\mathbb{E}_{I_i} \Omega + \min\{\rho_i^{(d)}, \mathbf{t}_i^{(d)}\}$, where $\hat{\mathbf{t}}_i^{(d)}$ denotes the $d$-th element of $\mathbf{t}_i$ and $\hat{\rho}_i^{(d)} = \max\{\rho_i^{(d)}, \mathbf{t}_i^{(d)}\}$. We used non-negative values of $\hat{\rho}_i^{(d)}$ and $|\mathbf{t}_i^{(d)}|$, because vectors $\rho_i$ and $\mathbf{t}_i$ may have negative elements.

Secondly, in addition to the accuracy of the rationale, we also measured the information loss of using the decision tree to represent a CNN, as a supplementary evaluation. A metric is the classification accuracy. Because $\hat{h}(x_i)$ denotes the prediction of $\hat{y}_i$ based on the best child in the second tree layer, we regarded $\hat{h}(\cdot)$ as the output of the tree and we evaluated the discrimination power of $\hat{h}(\cdot)$. We used values of $\hat{h}(x_i)$ for classification and compared its classification accuracy with the accuracy of the CNN. Another metric, namely the prediction error, measures the error of the
Table 1. Average number of nodes in the 2nd, 5th, 10th, 50th, and 100th layer of decision trees learned for VGG-16 nets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>2nd layer</th>
<th>5th layer</th>
<th>10th layer</th>
<th>50th layer</th>
<th>100th layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILSVRC Animal-Part</td>
<td>4.8</td>
<td>31.6</td>
<td>69.1</td>
<td>230.5</td>
<td>402.1</td>
</tr>
<tr>
<td>VGG Part</td>
<td>3.8</td>
<td>25.7</td>
<td>59.0</td>
<td>219.5</td>
<td>361.5</td>
</tr>
<tr>
<td>CUB200-2011</td>
<td>5.0</td>
<td>32.0</td>
<td>64.0</td>
<td>230.0</td>
<td>430.0</td>
</tr>
</tbody>
</table>

Table 2. Errors of object-part contributions that were estimated using nodes in the 2nd/5th/10th/50th/100th layer of the decision tree. The CNN was learned using the CUB200 dataset.

<table>
<thead>
<tr>
<th>layer</th>
<th>breast</th>
<th>forehead</th>
<th>nape</th>
<th>tail</th>
<th>average</th>
</tr>
</thead>
<tbody>
<tr>
<td>2nd</td>
<td>0.028</td>
<td>0.004</td>
<td>0.013</td>
<td>0.005</td>
<td>0.013</td>
</tr>
<tr>
<td>5th</td>
<td>0.024</td>
<td>0.004</td>
<td>0.010</td>
<td>0.006</td>
<td>0.011</td>
</tr>
<tr>
<td>10th</td>
<td>0.022</td>
<td>0.004</td>
<td>0.010</td>
<td>0.005</td>
<td>0.010</td>
</tr>
<tr>
<td>50th</td>
<td>0.018</td>
<td>0.003</td>
<td>0.008</td>
<td>0.005</td>
<td>0.009</td>
</tr>
<tr>
<td>100th</td>
<td>0.019</td>
<td>0.003</td>
<td>0.008</td>
<td>0.005</td>
<td>0.009</td>
</tr>
</tbody>
</table>

Table 3. Average fitness of contribution distributions based on nodes in the 2nd/5th/10th/50th/100th layer and leaf nodes, which reflects the accuracy of the estimated rationale of a prediction.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>2nd</th>
<th>5th</th>
<th>10th</th>
<th>50th</th>
<th>100th</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG-16 VOC Part</td>
<td>0.22</td>
<td>0.30</td>
<td>0.36</td>
<td>0.53</td>
<td>0.67</td>
</tr>
<tr>
<td>VGG-16 CUB200-2011</td>
<td>0.21</td>
<td>0.26</td>
<td>0.28</td>
<td>0.33</td>
<td>0.37</td>
</tr>
<tr>
<td>VGG-M VOC Part</td>
<td>0.35</td>
<td>0.38</td>
<td>0.46</td>
<td>0.61</td>
<td>0.78</td>
</tr>
<tr>
<td>VGG-M CUB200-2011</td>
<td>0.44</td>
<td>0.44</td>
<td>0.46</td>
<td>0.59</td>
<td>0.63</td>
</tr>
<tr>
<td>AlexNet VOC Part</td>
<td>0.33</td>
<td>0.35</td>
<td>0.41</td>
<td>0.63</td>
<td>0.80</td>
</tr>
<tr>
<td>AlexNet CUB200-2011</td>
<td>0.40</td>
<td>0.40</td>
<td>0.43</td>
<td>0.48</td>
<td>0.52</td>
</tr>
</tbody>
</table>

Table 4. Average classification accuracy and average prediction error based on nodes in the 2nd/5th/10th/50th/100th layer and leaf nodes of the tree. The classification accuracy and the prediction error reflect the accuracy of the estimated rationale of a prediction.

<table>
<thead>
<tr>
<th>CNN</th>
<th>Classification accuracy</th>
<th>Prediction error</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG-16 VOC Part</td>
<td>95.4</td>
<td>94.2</td>
</tr>
<tr>
<td>VGG-16 CUB200-2011</td>
<td>96.5</td>
<td>91.5</td>
</tr>
<tr>
<td>VGG-M VOC Part</td>
<td>95.2</td>
<td>94.2</td>
</tr>
<tr>
<td>VGG-M CUB200-2011</td>
<td>96.0</td>
<td>97.2</td>
</tr>
<tr>
<td>AlexNet VOC Part</td>
<td>95.5</td>
<td>92.7</td>
</tr>
<tr>
<td>AlexNet CUB200-2011</td>
<td>95.8</td>
<td>94.9</td>
</tr>
</tbody>
</table>

6. Conclusion and discussions

In this study, we use a decision tree to explain CNN predictions at the semantic level. We have developed a method to revise a CNN and build a tight coupling of the CNN and a decision tree. The proposed decision tree encodes decision modes of the CNN as quantitative rationales for each CNN prediction. Our method does not need any annotations of object parts or textures in training images to guide the learning the CNN. We have tested our method in different benchmark datasets, and experiments have proved the effectiveness of our approach.

Note that theoretically, the decision tree only provides an approximate explanation for CNN predictions, instead of a precise reconstruction of CNN representation details. There are two reasons. Firstly, without accurate object-part annotations to supervise the learning of CNNs, the filter loss can only roughly make each filter to represent an object part. The filter may be activated by unrelated visual concepts in a few challenging images. Secondly, the decision mode in each node ignores insignificant object-part filters to ensure a sparse representation of the decision mode.

Acknowledgements

This work is partially funded by Microsoft Research Asia, Huawei Key Laboratory of Shanghai Jiao Tong University, DARPA XAI Award N66001-17-2-4029, NSF IIS 1423305, and ARO project W911NF1810296.
References


[34] N. Wolchover. New theory cracks open the black box of deep learning. In Quanta Magazine, 2017. 1


