Explaining Neural Networks Semantically and Quantitatively

Runjin Chen*, Hao Chen*, Jie Ren, Ge Huang, and Quanshi Zhang†
Shanghai Jiao Tong University

Abstract

This paper presents a method to pursue a semantic and quantitative explanation for the knowledge encoded in a convolutional neural network (CNN). The estimation of the specific rationale of each prediction made by the CNN presents a key issue of understanding neural networks, and it is of significant values in real applications. In this study, we propose to distill knowledge from the CNN into an explainable additive model, which explains the CNN prediction quantitatively. We discuss the problem of the biased interpretation of CNN predictions. To overcome the biased interpretation, we develop prior losses to guide the learning of the explainable additive model. Experimental results have demonstrated the effectiveness of our method.

1. Introduction

Convolutional neural networks (CNNs) [20, 18, 13] have achieved superior performance in various tasks. Besides the discrimination power of neural networks, the interpretability of networks has received an increasing attention in recent years. The network interpretability is directly related to the trustworthiness of a CNN, which is crucial in critical applications. As discussed in [42], a high testing accuracy cannot fully ensure that the CNN encodes correct logic. Instead, a CNN may make predictions using unreliable reasons.

In this paper, we focus on the post-hoc explanation of a well trained CNN, instead of learning a new model with interpretable representations. Previous studies usually interpreted neural networks at the pixel level, such as the network visualization [40, 25, 30, 8, 10, 28] and the extraction of pixel-level attribution/importance maps [17, 26, 23].

In contrast to above qualitative explanation of CNNs, semantically and quantitatively clarifying the decision-making logic of each network prediction presents a more trustworthy way to diagnose neural networks. Fig. 1 compares our explanation with previous studies.

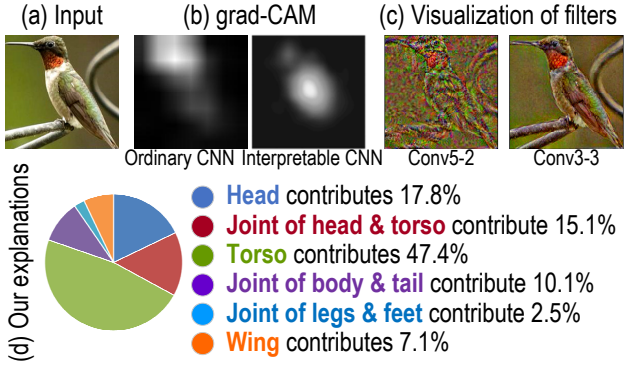

Figure 1. Different types of explanations for CNNs. We compare (d) our task of quantitatively and semantically explaining CNN predictions with previous studies of interpreting CNNs, such as (b) the grad-CAM [28] and (c) CNN visualization [25]. Given an input image (a) Our method generates a report, which quantitatively explains which object parts activate the CNN and how much these parts contribute to the prediction.

- *Semantic explanations:* This study aims to explain the logic of each network prediction using clear visual concepts, instead of using intermediate-layer features without clear meanings or using raw pixels. For example, our method estimates the numerical contribution of specific attributes or object parts to each prediction. Semantic explanations satisfy specific demands in real applications.
- *Quantitative explanations:* Unlike qualitative explanations or visualization of neural networks [40, 25, 8, 28], our method decomposes the overall prediction score into scores of compositional visual concepts, which ensures strict quantitative explanations of the prediction.

Quantitative explanations enable people to accurately diagnose feature representations inside neural networks and help neural networks earn trust from people. As shown in Figs. 5 and 7, we analyze the quantitative rationale of each prediction, *i.e.* clarifying which visual concepts activate the CNN and how much they contribute to the prediction. Abnormal explanations, which conflict with people's common sense, usually reflect problematic feature representations.

Above "semantic explanations" and "quantitative explanations" are of considerable values in real applications. Quantitatively decomposing the prediction score into val-

*Ruijin Chen, Jie Ren, Ge Huang, and Quanshi Zhang are with the John Hopcroft Center, Shanghai Jiao Tong University. This research was done when Hao Chen was an internship student at the Shanghai Jiao Tong University. Runjin Chen and Hao Chen contribute equally to this research.
†Quanshi Zhang is the corresponding author.

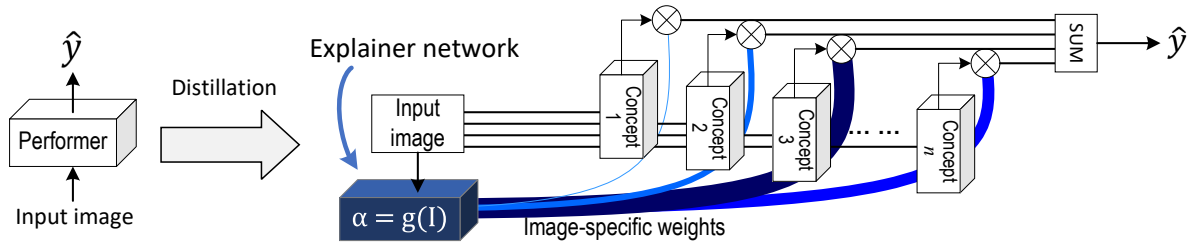

Figure 2. Task. We distill knowledge of a performer into an explainer as a paraphrase of the performer's representations. The explainer decomposes the output score into value components of semantic concepts, thereby obtaining semantic explanations for the performer.

ue components of clear visual concepts represents one of core challenges of understanding neural networks.

Method: We are given a pre-trained CNN. In this study, we propose to learn another neural network, namely an *explainer* network, in order to explain the specific rationale of each CNN prediction semantically and quantitatively. Accordingly, the target CNN is termed a *performer* network.

The explainer uses a set of pre-trained visual concepts to explain the performer's prediction. People can manually define the set of visual concepts for explanation. People can either use off-the-shelf models to represent these visual concepts, or train new models for them. Given trained models of these visual concepts, the learning of the explainer can be conducted without any additional image annotations.

The explainer uses trained visual concepts to mimic the decision-making logic inside the performer and generate similar prediction scores. As shown in Fig. 2, the explainer is designed as an additive model, which decomposes the prediction score into the sum of multiple value components of visual concept. Note that

- The explainer is **NOT** a linear model. The explainer $g(I)$ estimates weights for different visual concepts that are specific to each input image I.
- We can roughly consider each value component as a visual concept's contribution to the prediction score.
- Although the explainer $g(I)$ is a black-box model, the formulation of adding values of visual concepts to approximate the performer's prediction score still ensures the high transparency of the explanation.

We learn the explainer via knowledge distillation without any human annotations as additional supervision. It is because the explainer needs to objectively reflect the performer's logic, instead of encoding subjective human annotations and generating seemingly good explanations.

In this way, the explainer can be regarded as a semantic paraphrase of the performer. Theoretically, manually selected visual concepts usually cannot represent all logic of the performer. The prediction score, which cannot be explained by the visual concepts, also need to be analyzed. Thus, we quantify the difference of the prediction between the performer and the explainer, in order to disentangle the explainable and unexplainable information in the performer.

Post-hoc explaining black-box networks or learning interpretable networks: Explaining black-box models has much broader applicability than learning new interpretable networks. It is because (i) interpretable networks usually have specific requirements for structures [27] or losses [43], which limit the model flexibility; (ii) the model interpretability is not equivalent to, and usually even conflicts with the discrimination power of the model [43, 27].

Compared to forcing the performer to learn interpretable features, our method explains the performer without affecting the discrimination power of the performer. The coupling of the performer and the explainer solves the dilemma between the interpretability and the discriminability.

Core challenges: Distilling knowledge from a pre-trained neural network into an additive model may yield biased interpretation of the performer's predictions. *I.e.* when we use a large number of visual concepts to explain the logic inside the performer, the explainer may biasedly select very few visual concepts, instead of all visual concepts, as the rationale of the prediction (see Fig. 6). Just like the over-fitting problem, theoretically, the biased interpretation may be caused by the over-parameterizations of the explainer and is an ill-defined problem. Therefore, we propose a new loss for prior weights of visual concepts to overcome the bias-interpreting problem. The loss forces the explainer to compute a similar Jacobian of the prediction score *w.r.t.* visual concepts as the performer in early epochs.

Contributions of this study are summarized as follows. (i) In this study, we focus on a new explanation strategy, *i.e.* semantically and quantitatively explaining CNN predictions. (ii) We propose to distill knowledge from a pre-trained performer into an interpretable additive explainer for explanation. We develop novel losses to overcome the typical bias-interpreting problem. Our explanation strategy does not hurt the discrimination power of the performer. (iii) Theoretically, the proposed method is a generic solution to explaining neural networks. We have applied our method to different benchmark CNNs for different applications, which has proved the broad applicability of our method.

2. Related work

In this paper, we limit our discussion within the scope of understanding feature representations of neural networks.

Network visualization: The visualization of feature

representations inside a neural network is the most direct way of opening the black-box of the neural network. Related techniques include gradient-based visualization [40, 25, 30, 39] and up-convolutional nets [8] to invert feature maps of conv-layers into images. However, recent visualization results with clear semantic meanings were usually generated with strict constraints. These constraints made visualization results biased towards people's preferences. Subjectively visualizing all information of a filter usually produced chaotic results. Thus, there is still a considerable gap between network visualization and semantic explanations for neural networks.

Network diagnosis: Some studies diagnose feature representations inside a neural network. [38] measured features transferability in intermediate layers of a neural network. [1] visualized feature distributions of different categories in the feature space. [26, 23, 16, 10, 28] extracted rough pixel-level correlations between network inputs and outputs, *i.e.* estimating image regions that directly contribute the network output. Zhang *et al.* [24, 12] and Chen *et al.* [4] used the mutual information between the input and the output/intermediate-layer feature to quantify the information flow inside the network and extract important input units. Network-attack methods [17, 33] computed adversarial samples to diagnose a CNN. [19] discovered knowledge blind spots of a CNN in a weakly-supervised manner. [42] examined representations of conv-layers and automatically discover biased representations of a CNN due to the dataset bias. However, above methods usually analyzed a neural network at the pixel level and did not summarize the network knowledge into clear visual concepts.

[2] defined six types of semantics for CNN filters, *i.e.* objects, parts, scenes, textures, materials, and colors. Then, [46] proposed a method to compute the image-resolution receptive field of neural activations in a feature map. Fong and Vedaldi [9] analyzed how multiple filters jointly represented a certain semantic concept. Other studies retrieved intermediate-layer features from CNNs representing clear concepts. [41] mined object-part features from intermediate layers of a CNN. [41] used an explanatory graph to represent the semantic hierarchy between object-part features of different layers. [29] retrieved features to describe objects from feature maps, respectively. [46, 47] selected neural units to describe scenes. Note that strictly speaking, each CNN filter usually represents a mixture of multiple semantic concepts. Unlike previous studies, we are more interested in analyzing the quantitative contribution of each semantic concept to each prediction, which was not discussed in previous studies.

Learning interpretable representations: A new trend in the scope of network interpretability is to learn interpretable feature representations in neural networks [15, 32, 21] in an un-/weakly-supervised manner. Capsule nets [27] and interpretable RCNN [37] learned interpretable features in intermediate layers. InfoGAN [5] and β-VAE [14] learned well-disentangled codes for generative networks. Interpretable CNNs [43] learned filters in intermediate layers to represent object parts without given part annotations. However, as mentioned in [2], interpretable features usually do not have a high discrimination power. Therefore, we use the explainer to interpret the pre-trained performer without hurting the discriminability of the performer.

Explaining neural networks via knowledge distillation: Distilling knowledge from a black-box model into an explainable model is an emerging direction in recent years. In contrast, we pursue the explicitly quantitative explanation for each CNN prediction. [7] learned an explainable additive model, and [35] distilled knowledge of a network into an additive model. In order to disentangle feature representations of object parts from intermediate layers of a CNN, [44] distilled the CNN's knowledge into an explainer network with interpretable conv-layers, in which each filter represented a specific object part. [11, 34, 3, 36] distilled representations of neural networks into tree structures. These methods did not explain the network knowledge using human-interpretable semantic concepts. Zhang *et al.* [45] used a tree structure to approximately summarize the rationale of CNN predictions into generic decision-making logics. Compared to previous additive models [35], our research successfully overcomes the bias-interpreting problem, which is the core challenge when there are lots of visual concepts for explanation.

3. Algorithm

In this section, we distill knowledge from a pre-trained performer f to an explainable additive model. We are given a performer f and n models $\{f_i|i = 1, 2, \ldots, n\}$ that are pre-trained to detect n visual concepts. The n neural networks may share features in low layers with the performer. We are also given training samples for the performer f. We learn the explainer to use inference values of the n visual concepts to mimic the logic of the performer. We do not need any annotations on training samples *w.r.t.* the task, because additional supervision will push the explainer towards subjective annotations, instead of objective explanations for the performer.

Let $\hat{y} = f(I)$ denote the output of the performer for an input image I. For the performer with multiple outputs (*e.g.* for multi-category classification), we can learn an individual explainer to interpret each scalar output. Thus, in this study, we assume that $\hat{y}$ is a scalar without loss of generality. In particular, if the performer uses a softmax layer as the last layer, we use the feature score before the softmax layer as $\hat{y}$, so that $\hat{y}$'s neighboring scores will not affect $\hat{y}$. In our experiments, we only considered the classification task and took a single feature dimension before the softmax layer as

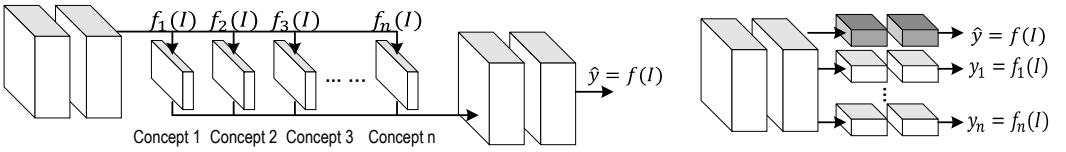

Figure 3. Two typical types of neural networks. (left) A performer models interpretable visual concepts in its intermediate layers. For example, each filter in a certain conv-layer represents a specific visual concept. (right) The performer and visual concepts are jointly learned, and they share features in intermediate layers.

$\hat{y}$.

We design the following additive explainer model, which uses a mixture of visual concepts to approximate the function of the performer. The explainer decomposes the prediction score $\hat{y}$ into value components of pre-defined visual concepts.

$$\begin{aligned} &\hat{y} \approx g_{\boldsymbol{\theta}}(I)^\top \mathbf{y} + b = b + \sum\nolimits_i \alpha_i \cdot y_i \quad \begin{matrix}\text{Quantitative contribution} \\ \text{of the } i\text{-th visual concept}\end{matrix} \\ &\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_n]^\top = g_{\boldsymbol{\theta}}(I) \\ &\mathbf{y} = [y_1, y_2, \ldots, y_n]^\top, \qquad y_i = f_i(I), \quad i = 1, 2, \ldots, n \end{aligned} \tag{1}$$

where y_i is given as the confidence of the detection of the i-th visual concept. α_i denotes the weight. b is a bias term.

- The additive form $g_{\boldsymbol{\theta}}(I)^\top \mathbf{y} + b$ does **NOT** represent a linear model, because different input images may obtain different weights $\boldsymbol{\alpha}$, which correspond to different decision-making logic of the performer. For example, a performer may mainly use head concept to classify a standing bird, while it may increase the weight for the wing concept to classify a flying bird.

Therefore, we design the explainer network $g_{\boldsymbol{\theta}}$ with parameters $\boldsymbol{\theta}$ (*i.e.* the explainer), which uses the input image I to estimate the n weights.

- We can regard the value of $\alpha_i \cdot y_i$ as the quantitative contribution of the i-th visual concept to the final prediction.
- The difference between predictions of the explainer and the performer $\hat{y} - (g_{\boldsymbol{\theta}}(I)^\top \mathbf{y} + b)$ reflects the limit of the representation capacity of visual concepts.

Therefore, the core task of this study is to learn the explainer $g_{\boldsymbol{\theta}}$ that estimates image-specific weights $\boldsymbol{\alpha} = g_{\boldsymbol{\theta}}(I)$ to mimic the performer. The loss for knowledge distillation is given as follows.

$$L = \|\hat{y} - g_{\boldsymbol{\theta}}(I)^\top \mathbf{y} - b\|^2 \tag{2}$$

However, without any prior knowledge about the weight α_i, the learning of $g_{\boldsymbol{\theta}}$ usually suffers from the bias-interpreting problem. The explainer g may biasedly select very few visual concepts to approximate the performer as a shortcut solution, instead of sophisticatedly learning relationships between the performer output and all visual concepts.

Thus, to overcome the bias-interpreting problem, we use a loss $\mathcal{L}$ for priors of $\boldsymbol{\alpha}$ to guide the learning in early epochs.

$$\begin{aligned} &\min_{\boldsymbol{\theta}, b} Loss, \qquad Loss = L + \lambda(t) \cdot \mathcal{L}(\boldsymbol{\alpha}, \mathbf{w}), \\ &\text{s.t.} \lim_{t \to \infty} \lambda(t) = 0 \end{aligned} \tag{3}$$

where $\mathbf{w}$ denotes prior weights, which represent a rough relationship between the performer's prediction value and n visual concepts. Just like $\boldsymbol{\alpha}$, different input images also have different prior weights $\mathbf{w}$. The loss $\mathcal{L}(\boldsymbol{\alpha}, \mathbf{w})$ penalizes the dissimilarity between $\boldsymbol{\alpha}$ and $\mathbf{w}$.

Note that prior weights $\mathbf{w}$ are approximated with strong assumptions (we will introduce two different ways of computing $\mathbf{w}$ later). We use inaccurate $\mathbf{w}$ to avoid significant biased interpretation, rather than pursue a high accuracy. Thus, we set a decreasing weight for $\mathcal{L}$, *i.e.* $\lambda(t) = \frac{\beta}{\sqrt{t}}$, where β is a scalar constant, and t denotes the epoch number. In this way, we mainly apply the prior loss $\mathcal{L}$ in early epochs. Then, in late epochs, the influence of $\mathcal{L}$ gradually decreases, and our method gradually shifts its attention to the distillation loss to boost the distillation quality.

3.1. Prior weights $\mathbf{w}$

In this subsection, we will discuss two typical types of weights $\mathbf{w}$ that are widely used in real applications.

Type 1: In some applications, the visual concept should be positively related to the prediction of the performer, *i.e.* each weight α_i must be a positive scalar (if the concept y_i is negatively related to the prediction, then the concept $-y_i$ is positively related to the prediction). For example, the detection score of the head part must be positively related to the detection of the entire animal. In this case, we use the cross-entropy between $\boldsymbol{\alpha}$ and $\mathbf{w}$ as the prior loss.

$$\mathcal{L}(\boldsymbol{\alpha}, \mathbf{w}) = crossEntropy(\frac{\boldsymbol{\alpha}}{\|\boldsymbol{\alpha}\|_1}, \frac{\mathbf{w}}{\|\mathbf{w}\|_1}) \tag{4}$$

where $\|\cdot\|_1$ denotes the L-1 norm. To ensure $\alpha_i \geq 0$, we add a non-linear activation layer $\boldsymbol{\alpha} = \log[1+\exp(x)]$ as the last layer of $g_{\boldsymbol{\theta}}$, where x is the output of the last conv-layer.

Type 2: In other applications without specific requirements for prior weights, the MSE loss between $\boldsymbol{\alpha}$ and $\mathbf{w}$ is used as the loss.

$$\mathcal{L}(\boldsymbol{\alpha}, \mathbf{w}) = \|\frac{\boldsymbol{\alpha}}{\|\boldsymbol{\alpha}\|_2} - \frac{\mathbf{w}}{\|\mathbf{w}\|_2}\|_2^2 \tag{5}$$

where $\|\cdot\|_2$ denotes the L-2 norm.

3.2. Typical cases for visual concepts

In this subsection, we will discuss two typical cases for visual concepts in real applications.

Case 1, features in intermediate layers of the performer are interpretable: As shown in Fig. 3(left), learning a neural network with interpretable features is an emerging research direction in recent years. For example, [43]

Figure 4. Visualization of interpretable filters in the top conv-layer of a CNN (Case 1), which were learned based on [43]. We projected activation regions on the feature map of the filter onto the image plane for visualization. Each filter represented a specific object part through different images.

proposed a method to learn CNNs for object classification, where each filter in a high conv-layer is exclusively triggered by the appearance of a specific object part (see Fig. 4).

Thus, the classification score of an object can be represented as a linear combination of elementary scores of object parts. Because such interpretable filters are automatically learned without part annotations, the quantitative explanation for the performer can be divided into the following two tasks: (i) annotating the part name of each filter, and (ii) learning an explainer to disentangle the exact additive contribution of each filter (or each object part).

In this way, each f_i is given as an interpretable filter of the performer. According to [42], we can roughly represent the network prediction as $\hat{y} \approx \sum_i w_i y_i + b$, where $y_i = \sum_{h,w} x_{hwi}$ and $w_i = \frac{1}{Z}\sum_{h,w} \frac{\partial \hat{y}}{\partial x_{hwi}}$. x_{hwi} is referred to as the activation unit in the location (h, w) of the i-th channel of the feature map $x \in \mathbb{R}^{H\times W\times n}$ of the interpretable conv-layer. y_i measures the confidence of the object part *w.r.t.* the i-th filter. Here, we roughly use the Jacobian of the network output *w.r.t.* the filter to approximate prior weights $\mathbf{w}$. Considering the normalization operation in Equation (4), prior weights $\mathbf{w}$ can be directly used to compute the first type of the loss without knowing the exact value of Z.

Case 2, neural networks for visual concepts share features in intermediate layers with the performer: As shown in Fig. 3(right), when a neural network is learned for multiple visual concepts, then people can directly explain a certain concept $\hat{y}$ by using all other concepts $\{y_i\}$. All these visual concepts share features in intermediate layers.

In this case, we estimate a rough numerical relationship between $\hat{y}$ and the score of each visual concept y_i. Let x be an intermediate-layer feature shared by both the target concept and the i-th visual concept.

When we modify the feature x, the change of y_i can be represented using a Taylor series $\Delta y_i = \frac{\partial y_i}{\partial x} \otimes \Delta x + O(\Delta^2 x)$, where $\otimes$ denotes the convolution operation. Thus, when we boost y_i by push x by $\Delta x = \epsilon \frac{\partial y_i}{\partial x}$ (ϵ is a small constant), the change of y_i can be approximated as $\Delta y_i = \epsilon \| \frac{\partial y_i}{\partial x} \|_F^2$, where $\| \cdot \|_F$ denotes the Frobenius norm. Meanwhile, Δx also affects the target concept by $\Delta \hat{y} = \epsilon \frac{\partial \hat{y}}{\partial x} \otimes \frac{\partial y_i}{\partial x}$. Thus, we can roughly estimate the weight as $w_i = \frac{\Delta \hat{y}}{\Delta y_i}$.

4. Experiments

We designed two experiments to use our explainers to diagnose different benchmark CNNs oriented to two different applications, in order to demonstrate the broad applicability of our method. In Experiment 1, we used the detection of object parts to explain the detection of the entire object. In Experiment 2, we used various face attributes to explain the prediction of another face attribute. We evaluated both the correctness of the explanation and the limit of the representation capacity of the explainer.

4.1. Experiment 1: using object parts to explain object classification

In this experiment, we used the method proposed in [43] to learn a CNN, where each filter in the top conv-layer represents a specific object part. We considered the CNN as a performer and regarded its interpretable filters in the top conv-layer as visual concepts to interpret the classification score. We followed standard experimental settings in [43], which used the Pascal-Part dataset [6] to learn six CNNs for the six animal[1] categories in the dataset. Each CNN was learned to classify a target animal from random images.

In addition, when the CNN had been learned, we further annotated the object-part name corresponding to each filter based on visualization results (see Fig. 4 for examples). We just annotated each filter of the top conv-layer in a performer once, so the total annotation cost was $O(N)$, where N is the filter number. Consequently, we assigned contributions of filters to its corresponding part, *i.e.* $Contri_p = \sum_{i\in\Omega_p} \alpha_i y_i$, where Ω_p denotes the set of filter indexes that were assigned to the part p.

Four types of CNNs as performers: Following experimental settings in [43], we applied our method to four types of CNNs, including the AlexNet [18], the VGG-M, the VGG-S, and the VGG-16 [31], *i.e.* we learned CNNs for six categories based on each network structure. Note that as discussed in [43], skip connections in residual networks increased the difficulty of learning part features, so they did not learn interpretable filters in residual networks.

Learning the explainer: The AlexNet/VGG-M/VGG-S/VGG-16 performer had 256/512/512/512 filters in its top

[1]Previous studies [6, 43] usually selected animal categories to test part localization, because animals usually contain non-rigid parts, which present significant challenges for part localization.

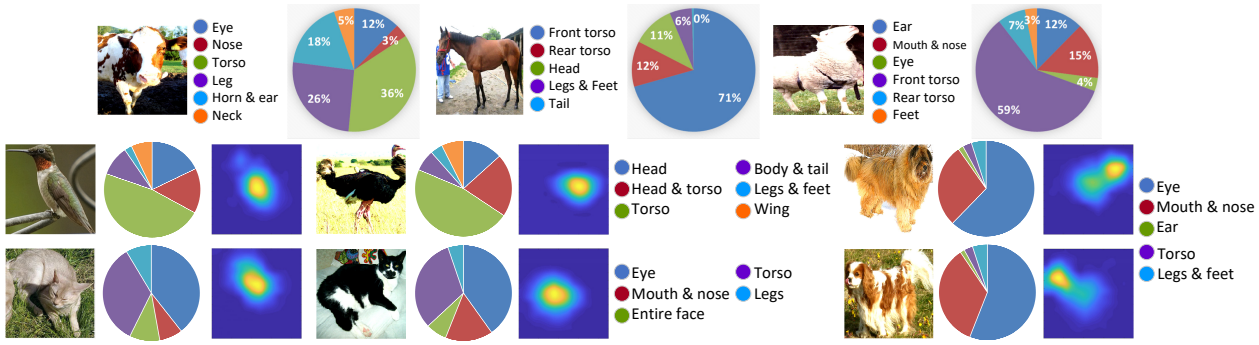

Figure 5. Quantitative explanations for the object classification made by performers. We annotated the part that was represented by each interpretable filter in the performer, and we assigned contributions of filters $\alpha_i y_i$ to object parts. Thus, each pie chart illustrates contributions of different object parts for a specific input image. All object parts made positive contributions to the classification score. Heatmaps correspond to the grad-CAM visualization [28] of feature maps of the performer to demonstrate the correctness of our explanations. Unlike pixel-level attention of the grad-CAM, our method explains the network at the semantic level.

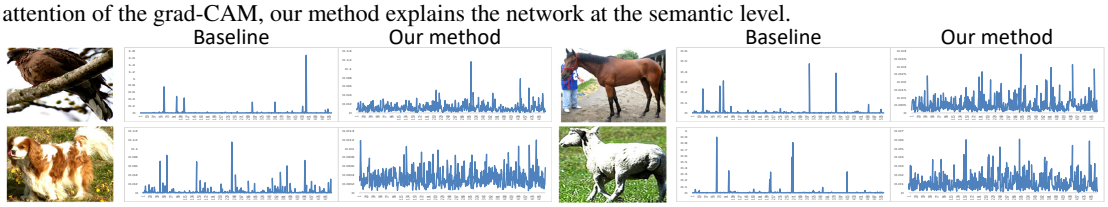

Figure 6. We compared the distribution of absolute contributions of different visual concepts (filters) that was estimated by our method and the distribution that was estimated by the baseline. The horizontal axis and the vertical axis denote the filter index and the contribution value, respectively. The baseline usually used very few visual concepts to make predictions, which was a typical case of biased interpretation.

conv-layer, so we set $n = 256/512/512/512$ for each network. We used the masked output of the top conv-layer as x to compute $\{y_i\}$ following the paradigm of Case 1. We used the 152-layer ResNet [13][2] as g to estimate weights of visual concepts[3]. We set $\beta = 10$ for the learning of all explainers. Note that all interpretable filters in the performer represented object parts of the target category on positive images, instead of describing random (negative) images. Intuitively, we needed to ensure a positive relationship between $\hat{y}$ and y_i. Thus, we filtered out negative prior weights $w_i \leftarrow \max\{w_i, 0\}$ and applied the cross-entropy loss in Equation (4) to learn the explainer.

Evaluation metric: The evaluation has two aspects, *i.e.* the correctness of the estimated explanations and the limit of the explanation capacity.

- *The correctness of the estimated explanations* for performer predictions is the first aspect of the evaluation. The first metric is the *error of the estimated contributions*. The explainer estimated numerical contributions of different visual concepts to the CNN prediction. For example, in Experiment 1, our method estimated the contribution of each annotated semantic part p, $Contri_p$. The error of the estimated contribution is computed as $\mathbb{E}_{I\in\mathbf{I}}[|Contri_p - y_p^*|]/\mathbb{E}_{I\in\mathbf{I}}[|y|]$, where y denotes the CNN output *w.r.t.* the image I; y_p^* is given as the ground-truth contribution of the part p. Let Δy_p denote the score change of y, when we removed all neural activations of filters corresponding the part p. Then, we computed $y_p^* = y\frac{\Delta y_p}{\sum_{p'}\Delta y_{p'}}$. In our experiments, we used semantic part annotations of the dog category to compute errors of the estimated contributions.

In addition, we used another metric, *i.e.* the *conditional entropy of the explanation*, for the fine-grained filter-level analysis of the correctness of the explanation. Let $c_i = \alpha_i y_i$ denote the estimated numerical contribution of the i-th visual concept, $\mathbf{c} = [c_1, c_2, \ldots, c_n]^\top$. We can represent the estimated contribution as $\mathbf{c}^* = \mathbf{c} + \boldsymbol{\epsilon}$, where $\mathbf{c}^* = \{c_i^*\}$ denotes the ground-truth contribution, and $\boldsymbol{\epsilon}$ is referred to as the residual that cannot be explained by the current model. The conditional entropy $H(\mathbf{c}^*|\mathbf{c})$ reflects the information loss using the estimated contributions for explanation. We assumed $\boldsymbol{\epsilon} \sim Gauss(\boldsymbol{\mu} = \mathbf{0}, \boldsymbol{\Sigma} = \sigma^2\mathbf{I})$. Thus, $p(\mathbf{c}^*|\mathbf{c}) = p(\boldsymbol{\epsilon})$ and $H(\mathbf{c}^*|\mathbf{c}) = H(\boldsymbol{\epsilon})$. For implementation, we used $\mathbf{y} \circ \mathbf{w}\frac{\|\boldsymbol{\alpha}\|}{\|\mathbf{w}\|}$ to approximate $\mathbf{c}^*$, where $\circ$ denotes the element-wise multiplication. We estimated the variance σ^2 from data to compute $H(\mathbf{c}^*|\mathbf{c})$. Besides, we assumed $\mathbf{c}^* \sim Gauss(\boldsymbol{\mu} = \mathbf{0}, \boldsymbol{\Sigma} = (\sigma^*)^2\mathbf{I})$ and computed the entropy of ground-truth explanations $H(\mathbf{c}^*)$ without additional information as a reference value. If the explainer successfully recovered most information of $\mathbf{c}^*$, then $H(\mathbf{c}^*|\mathbf{c})$ would be significantly lower than $H(\mathbf{c}^*)$.

Besides the quantitative evaluation, we also showed example explanations as a qualitative evaluation. As shown

[2]Considering the small size of the input feature map, we removed the first max-pooling layer and the last average-pooling layer.

[3]Note that the input of the ResNet was the feature map of the top conv-layer, rather than the original image I in experiments, so g can be considered as a cascade of conv-layers in the AlexNet/VGGs and the ResNet.

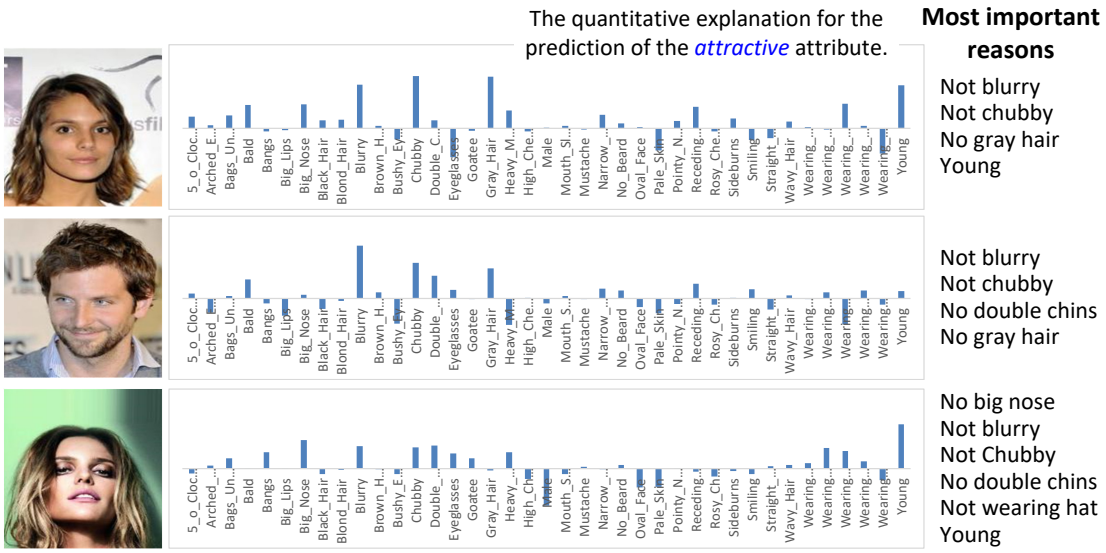

Figure 7. Quantitative explanations for face-attribution predictions made by performers. Bars indicate elementary contributions $\alpha_i y_i$ from features of different face attributes, rather than prediction values y_i of these attributes. For example, the network predicts a negative *goatee* attribute $y_{\text{goatee}} < 0$, and this information makes a positive contribution to the target *attractive* attribute, $\alpha_i y_i > 0$.

in Fig. 5, we used grad-CAM visualization [28] of feature maps to prove the correctness of our explanations.

- *The limit of the explanation capacity* also needs to be evaluated, *i.e.* we measured the performer information that could not be represented by the visual concepts. We proposed three metrics for evaluation. The first metric is the *prediction accuracy*. We compared the prediction accuracy of the performer with the prediction accuracy of using the explainer's output $y = g_{\boldsymbol{\theta}}(I)^\top \mathbf{y} + b$.

Another metric is the conditional entropy of the prediction. Let $\hat{y} = f(I)$ and $y = g_{\boldsymbol{\theta}}(I)^\top \mathbf{y} + b$ denote the ground-truth prediction and the estimated prediction, respectively. We can represent $\hat{y} = y + \epsilon_y$, where ϵ_y reflects the information that has not been encoded by the explainer. In this way, we used the conditional entropy $H(\hat{y}|y) = H(\epsilon_y)$ to measure the limit of the representation power of visual concepts. We used the original entropy $H(\hat{y})$ as a reference value. We assumed $\hat{y} \sim Gauss(\hat{\mu}, \hat{\sigma}^2)$ and $\epsilon_y \sim Gauss(\mu_\epsilon, \sigma_\epsilon^2)$ to compute $H(\hat{y})$ and $H(\hat{y}|y)$, where parameters $\hat{\mu}, \mu_\epsilon, \hat{\sigma}^2, \sigma_\epsilon^2$ were directly calculated using the data.

The third metric is the *relative deviation*, which measures a normalized output difference between the performer and the explainer. The relative deviation of the image I is normalized as $|\hat{y}_I - y_I| / (\max_{I' \in \mathbf{I}} \hat{y}_{I'} - \min_{I' \in \mathbf{I}} \hat{y}_{I'})$, where $\hat{y}_I$ and y_I denote predictions for the image I made by the performer and the explainer, respectively.

Above metrics reflected knowledge, which was not modeled by the explainer. Note that
(1) our objective was **not** to pursue an extremely low conditional entropy, because the limit of the representation power of visual concepts is an objective existence;
(2) compared to the representation power, our method paid more attention on the correctness of the explanation.

4.2. Experiment 2: explaining face attributes based on face attributes

In this experiment, we used the Celeba dataset [22] to learn a CNN based on the VGG-16 structure to estimate 40 face attributes. We selected a certain attribute as the target and used its prediction score as $\hat{y}$. Other 39 attributes were taken as visual concepts to explain the score of $\hat{y}$ ($n = 39$). The target attribute was selected from global attributes of the face, *i.e. attractive*, *heavy makeup*, *male*, and *young*. It is because global attributes can be described by local visual concepts, but the inverse is not. We learned an explainer for each target attribute. We used the same 152-layer ResNet structure as in Exp. 1 (expect for $n = 39$) as g to estimate weights. We followed the Case-2 implementation in Section 3.2 to compute prior weights $\mathbf{w}$, in which we used the 4096-dimensional output of the first fully-connected layer as the shared feature x. We set $\beta = 3000$ and used the L-2 norm loss in Equation (5) to learn all explainers. Evaluation metrics were the same as in Exp. 1.

4.3. Experimental results and analysis

We compared our method with the traditional baseline of only using the distillation loss to learn the explainer. Tables 3 and 4 evaluate the biased interpretation caused by explainers that were learned using our method and the baseline. Our method suffered much less from the bias-interpreting problem than the baseline. According to Tables 3 and 4, our method generated more informative explanations than the baseline. More crucially, Fig. 6 illustrates the distribution of absolute contributions of visual concepts,

	Experiment 1				Experiment 2				
	AlexNet	VGG-M	VGG-S	VGG-16	attractive	makeup	male	young	Avg.
Performer	93.9	94.2	95.5	95.4	81.5	92.3	98.7	88.3	90.2
Explainer	92.6	93.6	95.4	95.6	76.8	90.0	97.8	85.0	87.4

Table 1. Classification accuracy of the explainer and the performer. We used the decrease of the classification accuracy to measure the information that could not be explained by pre-defined visual concepts.

$H(\hat{y}\|y)$	Experiment 1				Experiment 2				
	AlexNet	VGG-M	VGG-S	VGG-16	attrac.	makeup	male	young	Avg.
$H(\hat{y})$	0.45	1.45	0.78	-1.54	0.11	0.80	1.33	0.28	0.63
Baseline	-1.62	0.15	-0.88	-3.94	-0.51	-0.50	-0.37	-0.35	-0.43
Our method	-2.33	-0.76	-1.20	-4.17	-0.33	0.05	-0.30	-0.38	-0.24

Table 2. Conditional entropy of $\hat{y}$ given the explainer's prediction, $H(\hat{y}|y)$. We used the conditional entropy to measure the information that could not be explained by pre-defined visual concepts. The entropy is defined on continuous variables, so it can be negative.

$H(\mathbf{c}^*\|\mathbf{c})$	Experiment 1				Experiment 2				
	AlexNet	VGG-M	VGG-S	VGG-16	attrac.	makeup	male	young	Avg.
$H(\mathbf{c}^*)$	-1.18	-1.71	-1.51	-3.74	1.62	1.00	0.83	1.20	1.16
Baseline	-1.35	-1.79	-1.43	-1.96	1.59	1.60	1.76	1.65	1.65
Our method	**-2.39**	**-2.34**	**-2.25**	**-4.79**	**1.22**	**0.35**	**-0.40**	**0.77**	**0.48**

Table 3. Conditional entropy of the explanation given the explainer's explanation, $H(\mathbf{c}^*|\mathbf{c})$. We used the conditional entropy to measure the inaccuracy of the explanation generated by the explainer. The entropy is defined on continuous variables, so it can be negative.

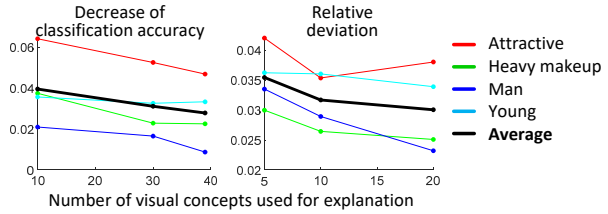

Figure 8. Explanation capacity of using different numbers of visual concepts for explanation. We used the relative deviation and the decrease of the classification error (*i.e.* the performer's accuracy minus the explainer's accuracy) using the explainer to roughly measure the limit of the explanation capacity. Using more visual concepts will increase the explanation capacity.

	eye	mouth & nose	ear	torso	leg	Avg.
Baseline	0.399	0.267	0.045	0.027	0.084	0.164
Ours ($\beta=10$)	0.202	**0.140**	**0.033**	0.032	0.092	0.100
Ours ($\beta=25$)	**0.181**	0.153	0.037	**0.019**	**0.053**	**0.089**

Table 4. Errors of the estimated object-part contributions. A lower error of our method indicates that the explanation yielded by our approach better fit the ground-truth rationale of a CNN prediction than the baseline.

i.e. the distribution of $\{|c_i|\}$, when we learned explainers using different methods. Part contributions estimated by our method better fit the ground truth than those estimated by the baseline. In contrast, the distillation baseline usually used very few visual concepts for explanation and ignored most strongly activated interpretable filters, which could be considered as biased interpretation.

Figs. 5 and 7 show examples of quantitative explanations for the prediction made by the performer. In particular, we also used the grad-CAM visualization [28] of feature maps of the performer to demonstrate the correctness of our explanations in Fig. 5. In addition, Tables 1 and 2 report the classification accuracy and the conditional entropy of the prediction to measure the representation capacity of visual concepts. Because our method used an additional loss to ensure the correctness of explanation, our method performed a bit worse in the regression of $\hat{y}$ than the baseline. Nevertheless, considering the high entropy of $H(\hat{y})$, our method still encoded most information of the performer.

Selection of visual concepts: How to select visual concepts for explanation is an important issue. The explanation capacity will decrease if related concepts are not selected for explanation. Fig. 8 evaluates the change of the explanation capacity, when we randomly selected different numbers of visual concepts to learn explainers for the estimation of face attributes. Note that during the computation of the relative deviation in Fig. 8, we set $\beta = \frac{0.1}{n}$ to remove effects of increasing the concept number for fair comparisons.

5. Conclusion and discussions

In this paper, we focus on a new explanation strategy, *i.e.* explaining the logic of each CNN prediction semantically and quantitatively, which presents considerable challenges in the scope of understanding neural networks. We propose to distill knowledge from a pre-trained performer into an interpretable additive explainer. We can consider that the performer and the explainer encode similar knowledge. The additive explainer decomposes the prediction score of the performer into value components from semantic visual concepts, in order to quantify contributions of different concepts. The strategy of using an explainer for explanation avoids decreasing the discrimination power of the performer. In preliminary experiments, we have applied our method to different benchmark CNN performers to prove its broad applicability.

Note that our objective is not to use pre-trained visual concepts to achieve super accuracy in classification/prediction. Instead, the explainer uses these visual concepts to mimic the logic of the performer.

In particular, biased interpretation is a big challenge of using an additive explainer to diagnose another neural network. We designed two losses to overcome the bias-interpreting problems. Besides, we measured the amount of the performer knowledge that could not be represented by the visual concepts in the explainer and used various metrics to evaluate the significance of biased interpretation.

References

[1] Mathieu Aubry and Bryan C. Russell. Understanding deep features with computer-generated imagery. *In ICCV*, 2015.

[2] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. *In CVPR*, 2017.

[3] Zhengping Che, Sanjay Purushotham, Robinder Khemani, and Yan Liu. Interpretable deep models for icu outcome prediction. *In American Medical Informatics Association (AMIA) Annual Symposium*, 2016.

[4] Jianbo Chen, Le Song, Martin J. Wainwright, and Michael I. Jordan. Learning to explain: An information-theoretic perspective on model interpretation. *In ICML*, 2018.

[5] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *In NIPS*, 2016.

[6] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts. *In CVPR*, 2014.

[7] Edward Choi, Mohammad Taha Bahadori, Joshua A. Kulas, Andy Schuetz, Walter F. Stewart, and Jimeng Sun. Retain: an interpretable predictive model for healthcare using reverse time attention mechansim. *in arXiv:1608.05745v4*, 2017.

[8] Alexey Dosovitskiy and Thomas Brox. Inverting visual representations with convolutional networks. *In CVPR*, 2016.

[9] Ruth Fong and Andrea Vedaldi. Net2vec: Quantifying and explaining how concepts are encoded by filters in deep neural networks. *In CVPR*, 2018.

[10] Ruth C. Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. *In arXiv:1704.03296v1*, 2017.

[11] Nicholas Frosst and Geoffrey Hinton. Distilling a neural network into a soft decision tree. *In arXiv:1711.09784*, 2017.

[12] Chaoyu Guan, Xiting Wang, Quanshi Zhang, Runjin Chen, Di He, and Xing Xie. Towards a deep and unified understanding of deep neural models in nlp. *In ICML*, 2019.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *In CVPR*, 2016.

[14] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. β-vae: learning basic visual concepts with a constrained variational framework. *In ICLR*, 2017.

[15] Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric P. Xing. Harnessing deep neural networks with logic rules. *In arXiv:1603.06318v2*, 2016.

[16] Pieter-Jan Kindermans, Kristof T. Schütt, Maximilian Alber, Klaus-Robert Müller, Dumitru Erhan, Been Kim, and Sven Dähne. Learning how to explain neural networks: Patternnet and patternattribution. *In ICLR*, 2018.

[17] PangWei Koh and Percy Liang. Understanding black-box predictions via influence functions. *In ICML*, 2017.

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *In NIPS*, 2012.

[19] Himabindu Lakkaraju, Ece Kamar, Rich Caruana, and Eric Horvitz. Identifying unknown unknowns in the open world: Representations and policies for guided exploration. *In AAAI*, 2017.

[20] Yann LeCun, Lèon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *In Proceedings of the IEEE*, 1998.

[21] Renjie Liao, Alex Schwing, Richard Zemel, and Raquel Urtasun. Learning deep parsimonious representations. *In NIPS*, 2016.

[22] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. *In ICCV*, 2015.

[23] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *In NIPS*, 2017.

[24] Haotian Ma, Yinqing Zhang, Fan Zhou, and Quanshi Zhang. Quantifying layerwise information discarding of neural networks. *In arXiv:1906.04109*, 2019.

[25] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. *In CVPR*, 2015.

[26] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?" explaining the predictions of any classifier. *In KDD*, 2016.

[27] Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. Dynamic routing between capsules. *In NIPS*, 2017.

[28] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *In ICCV*, 2017.

[29] Marcel Simon and Erik Rodner. Neural activation constellations: Unsupervised part model discovery with convolutional networks. *In ICCV*, 2015.

[30] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: visualising image classification models and saliency maps. *In arXiv:1312.6034*, 2013.

[31] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *In ICLR*, 2015.

[32] Austin Stone, Huayan Wang, Yi Liu, D. Scott Phoenix, and Dileep George. Teaching compositionality to cnns. *In CVPR*, 2017.

[33] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *In arXiv:1312.6199v4*, 2014.

[34] Sarah Tan, Rich Caruana, Giles Hooker, and Albert Gordo. Transparent model distillation. *In arXiv:1801.08640*, 2018.

[35] Joel Vaughan, Agus Sudjianto, Erind Brahimi, Jie Chen, and Vijayan N. Nair. Explainable neural networks based on additive index models. *in arXiv:1806.01933*, 2018.

[36] Mike Wu, Michael C. Hughes, Sonali Parbhoo, Maurizio Zazzi, Volker Roth, and Finale Doshi-Velez. Beyond sparsity: Tree regularization of deep models for interpretability. *In NIPS TIML Workshop*, 2017.

[37] Tianfu Wu, Xilai Li, Xi Song, Wei Sun, Liang Dong, and Bo Li. Interpretable r-cnn. *In arXiv:1711.05226*, 2017.

[38] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *In NIPS*, 2014.

[39] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *In ICML Deep Learning Workshop*, 2015.

[40] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *In ECCV*, 2014.

[41] Quanshi Zhang, Ruiming Cao, Feng Shi, Ying Nian Wu, and Song-Chun Zhu. Interpreting cnn knowledge via an explanatory graph. *In AAAI*, 2018.

[42] Quanshi Zhang, Wenguan Wang, and Song-Chun Zhu. Examining cnn representations with respect to dataset bias. *In AAAI*, 2018.

[43] Quanshi Zhang, Ying Nian Wu, and Song-Chun Zhu. Interpretable convolutional neural networks. *In CVPR*, 2018.

[44] Quanshi Zhang, Yu Yang, Yuchen Liu, Ying Nian Wu, and Song-Chun Zhu. Unsupervised learning of neural networks to explain neural networks. *in arXiv:1805.07468*, 2018.

[45] Quanshi Zhang, Yu Yang, Haotian Ma, and Ying Nian Wu. Interpreting cnns via decision trees. *In CVPR*, 2019.

[46] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. *In ICRL*, 2015.

[47] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. *In CVPR*, 2016.