EXPLAINING NEURAL NETWORKS SEMANTICALLY AND QUANTITATIVELY

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper presents a method to explain the knowledge encoded in a convolutional neural network (CNN) quantitatively and semantically. The analysis of the specific rationale of each prediction made by the CNN presents a key issue of understanding neural networks, but it is also of significant practical values in certain applications. In this study, we propose to distill knowledge from the CNN into an explainable additive model, so that we can use the explainable model to provide a quantitative explanation for the CNN prediction. We analyze the typical bias-interpreting problem of the explainable model and 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) (LeCun et al., 1998; Krizhevsky et al., 2012; He et al., 2016) have achieved superior performance in various tasks, such as object classification and detection. Besides the discrimination power of neural networks, the interpretability of neural networks has received an increasing attention in recent years.

In this paper, we focus on a new problem, *i.e.* explaining the specific rationale of each network prediction *semantically* and *quantitatively*. "Semantic explanations" and "quantitative explanations" are two core issues of understanding neural networks.

1. **Semantic explanations:** We hope to explain the logic of each network prediction using clear visual concepts, instead of using middle-layer features without clear meanings or simply extracting pixel-level correlations between network inputs and outputs. We believe that semantic explanations may satisfy specific demands in real applications.
2. **Quantitative explanations:** In contrast to traditional qualitative explanations for neural networks, quantitative explanations enable people to diagnose feature representations inside neural networks and help neural networks earn trust from people. We expect the neural network to provide the quantitative rationale of the prediction, *i.e.* clarifying which visual concepts activate the neural network and how much they contribute to the prediction score.

Above two requirements present significant challenges to state-of-the-art algorithms. To the best of our knowledge, no previous studies simultaneously explained network predictions using clear visual concepts and quantitatively decomposed the prediction score into value components of these visual concepts.

Task: Therefore, in this study, we propose to learn another neural network, namely an *explainer* network, to explain CNN predictions. Accordingly, we can call the target CNN a *performer* network. Besides the performer, we also require a set of models that are pre-trained to detect different visual concepts. These visual concepts will be used to explain the logic of the performer's prediction. We are also given input images of the performer, but we do not need any additional annotations on the images. Then, the explainer is learned to mimic the logic inside the performer, *i.e.* the explainer receives the same features as the performer and is expected to generate similar prediction scores.

As shown in Fig. 1, the explainer uses pre-trained visual concepts to explain each prediction. The explainer is designed as an additive model, which decomposes the prediction score into the sum of

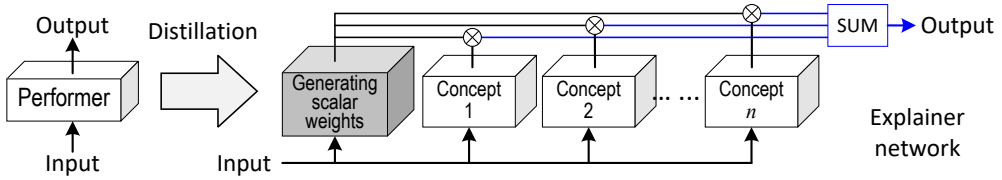

Figure 1: Explainer. We distill knowledge of a performer into an explainer as a paraphrase of the performer's representations. The explainer decomposes the prediction score into value components of semantic concepts, thereby obtaining quantitative semantic explanations for the performer.

multiple value components. Each value component is computed based on a specific visual concept. In this way, we can roughly consider these value components as quantitative contributions of the visual concepts to the final prediction score.

More specifically, we learn the explainer via knowledge distillation. Note that we do **not** use any ground-truth annotations on input images to supervise the explainer. It is because the task of the explainer is not to achieve a high prediction accuracy, but to mimic the performer's logic in prediction, no matter whether the performer's prediction is correct or not.

Thus, the explainer can be regarded as a semantic paraphrase of feature representations inside the performer, and we can use the explainer to understand the logic of the performer's prediction. Theoretically, the explainer usually cannot recover the exact prediction score of the performer, owing to the limit of the representation capacity of visual concepts. The difference of the prediction score between the performer and the explainer corresponds to the information that cannot be explained by the visual concepts.

Challenges: Distilling knowledge from a pre-trained neural network into an additive model usually suffers from the problem of bias-interpreting. 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 (Fig. 4 in the appendix visualizes the bias-interpreting problem). Just like the typical over-fitting problem, theoretically, the bias interpreting is an ill-defined problem. To overcome this problem, we propose two types of losses for prior weights of visual concepts to guide the learning process. The prior weights push the explainer to compute a similar Jacobian of the prediction score *w.r.t.* visual concepts as the performer in early epochs, in order to avoid bias-interpreting.

Originality: Our "semantic-level" explanation for CNN predictions has essential differences from traditional studies of "pixel-level" interpreting neural networks, such as the visualization of features in neural networks (Zeiler & Fergus, 2014; Mahendran & Vedaldi, 2015; Simonyan et al., 2013; Dosovitskiy & Brox, 2016; Fong & Vedaldi, 2017; Selvaraju et al., 2017), the extraction of pixel-level correlations between network inputs and outputs (Koh & Liang, 2017; Ribeiro et al., 2016; Lundberg & Lee, 2017), and the learning of neural networks with interpretable middle-layer features (Zhang et al., 2018b; Sabour et al., 2017).

In particular, the explainer explains the performer without affecting the original discrimination power of the performer. As discussed in (Bau et al., 2017), the interpretability of features is not equivalent to, and usually even conflicts with the discrimination power of features. Compared to forcing the performer to learn interpretable features, our strategy of explaining the performer solves the dilemma between the interpretability and the discriminability. In addition, our quantitative explanation has special values beyond the qualitative analysis of CNN predictions (Zhang et al., 2018c).

Potential values of the explainer: Quantitatively and semantically explaining a performer is of considerable practical values when the performer needs to earn trust from people in critical applications. As mentioned in (Zhang et al., 2018a), owing to the potential bias in datasets and feature representations, a high testing accuracy still cannot fully ensure correct feature representations in neural networks. Thus, semantically and quantitatively clarifying the logic of each network prediction is a direct way to diagnose feature representations of neural networks. Fig. 3 shows example explanations for the performer's predictions. Predictions whose explanations conflict people's common sense may reflect problematic feature representations inside the performer.

Contributions of this study are summarized as follows. (i) In this study, we focus on a new task, *i.e.* semantically and quantitatively explaining CNN predictions. (ii) We propose a new method to

explain neural networks, *i.e.* distilling knowledge from a pre-trained performer into an interpretable additive explainer. Our strategy of using the explainer to explain the performer avoids hurting the discrimination power of the performer. (iii) We develop novel losses to overcome the typical bias-interpreting problem. Preliminary experimental results have demonstrated the effectiveness of the proposed method. (iv) Theoretically, the proposed method is a generic solution to the problem of interpreting 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 (Zeiler & Fergus, 2014; Mahendran & Vedaldi, 2015; Simonyan et al., 2013; Yosinski et al., 2015) and up-convolutional nets (Dosovitskiy & Brox, 2016) 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. (Yosinski et al., 2014) measured features transferability in intermediate layers of a neural network. (Aubry & Russell, 2015) visualized feature distributions of different categories in the feature space. (Ribeiro et al., 2016; Lundberg & Lee, 2017; Kindermans et al., 2018; Fong & Vedaldi, 2017; Selvaraju et al., 2017) extracted rough pixel-level correlations between network inputs and outputs, *i.e.* estimating image regions that directly contribute the network output. Network-attack methods (Koh & Liang, 2017; Szegedy et al., 2014) computed adversarial samples to diagnose a CNN. (Lakkaraju et al., 2017) discovered knowledge blind spots of a CNN in a weakly-supervised manner. (Zhang et al., 2018a) 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.

(Bau et al., 2017) defined six types of semantics for CNN filters, *i.e.* objects, parts, scenes, textures, materials, and colors. Then, (Zhou et al., 2015) proposed a method to compute the image-resolution receptive field of neural activations in a feature map. Other studies retrieved middle-layer features from CNNs representing clear concepts. (Simon & Rodner, 2015) retrieved features to describe objects from feature maps, respectively. (Zhou et al., 2015; 2016) 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 (Hu et al., 2016; Stone et al., 2017; Liao et al., 2016) in an un-/weakly-supervised manner. Capsule nets (Sabour et al., 2017) and interpretable RCNN (Wu et al., 2017b) learned interpretable features in intermediate layers. InfoGAN (Chen et al., 2016) and β-VAE (Higgins et al., 2017) learned well-disentangled codes for generative networks. Interpretable CNNs (Zhang et al., 2018b) learned filters in intermediate layers to represent object parts without given part annotations. However, as mentioned in (Bau et al., 2017; Zhang et al., 2018c), 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. (Zhang et al., 2018d) used a tree structure to summarize the inaccurate[1] rationale of each CNN prediction into generic decision-making models for a number of samples. In contrast, we pursue the explicitly quantitative

[1]Please see the appendix for details

explanation for each CNN prediction. (Choi et al., 2017) learned an explainable additive model, and (Vaughan et al., 2018) distilled knowledge of a network into an additive model. (Frosst & Hinton, 2017; Tan et al., 2018; Che et al., 2016; Wu et al., 2017a) distilled representations of neural networks into tree structures. These methods did not explain the network knowledge using human-interpretable semantic concepts. More crucially, compared to previous additive models (Vaughan et al., 2018), 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 neural networks $\{f_i | i = 1, 2, \ldots, n\}$ that are pre-trained to detect n different visual concepts. We learn the n neural networks along with the performer, and the n neural networks are expected to share low-layer features with the performer. Our method also requires a set of training samples for the performer f. The goal of the explainer is to use inference values of the n visual concepts to explain prediction scores of the performer. Note that we do not need any annotations on training samples *w.r.t.* the task, because additional supervision will push the explainer towards a good performance of the task, instead of objectively reflecting the knowledge in the performer.

Given an input image I, let $\hat{y} = f(I)$ denote the output of the performer. Without loss of generality, we assume that $\hat{y}$ is a scalar. If the performer has multiple outputs (*e.g.* a neural network for multi-category classification), we can learn an explainer to interpret each scalar output of the performer. In particular, when the performer takes 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 the value of $\hat{y}$.

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.

$$\hat{y} \approx \underbrace{\alpha_1(I) \cdot y_1}_{\text{Quantitative contribution from the first visual concept}} + \alpha_2(I) \cdot y_2 + \ldots + \alpha_n(I) \cdot y_n + b, \quad y_i = f_i(I), \quad i = 1, 2, \ldots, n \tag{1}$$

where y_i and $\alpha_i(I)$ denote the scalar value and the weight for the i-th visual concept, respectively. b is a bias term. y_i is given as the strength or confidence of the detection of the i-th visual concept. We can regard the value of $\alpha_i(I) \cdot y_i$ as the quantitative contribution of the i-th visual concept to the final prediction. In most cases, the explainer cannot recover all information of the performer. The prediction difference between the explainer and the performer reflects the limit of the representation capacity of visual concepts.

According to the above equation, the core task of the explainer is to estimate a set of weights $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_n]$, which minimizes the difference of the prediction score between the performer and the explainer. Different input images may obtain different weights $\boldsymbol{\alpha}$, which correspond to different decision-making modes of the performer. For example, a performer may mainly use head patterns to classify a standing bird, while it may increase the weight for the wing concept to classify a flying bird. Therefore, we design another neural network g with parameters $\boldsymbol{\theta}_g$ (*i.e.* the explainer), which uses the input image I to estimate the n weights. We learn the explainer with the following knowledge-distillation loss.

$$\boldsymbol{\alpha} = g(I), \qquad L = \|\hat{y} - \sum_{i=1}^{n} \alpha_i \cdot y_i - b\|^2 \tag{2}$$

However, without any prior knowledge about the distribution of the weight α_i, the learning of g usually suffers from the problem of bias-interpreting. The neural network 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 process in early epochs.

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

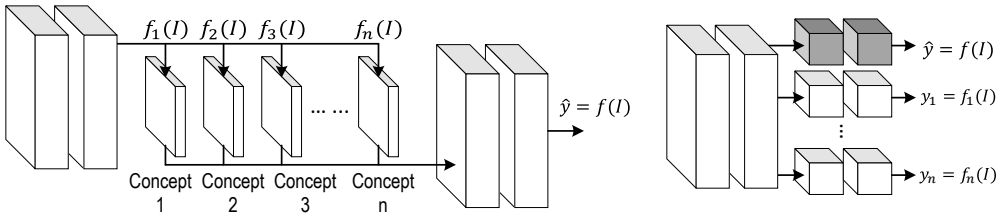

Figure 2: Two typical types of neural networks. (left) A performer models interpretable visual concepts in its intermediate layers. For example, each filter in a certain conv-layer represents a specific visual concept. (right) The performer and visual concepts are jointly learned, and they share features in intermediate layers.

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 the 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 bias-interpreting, 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 for a high distillation accuracy.

We design two types of losses for prior weights, as follows.

$$\mathcal{L}(\boldsymbol{\alpha}, \mathbf{w}) = \begin{cases} crossEntropy(\frac{\boldsymbol{\alpha}}{\|\boldsymbol{\alpha}\|_1}, \frac{\mathbf{w}}{\|\mathbf{w}\|_1}), & \forall i, \alpha_i, w_i \geq 0 \\ \| \frac{\boldsymbol{\alpha}}{\|\boldsymbol{\alpha}\|_2} - \frac{\mathbf{w}}{\|\mathbf{w}\|_2} \|_2^2, & \text{otherwise} \end{cases} \tag{4}$$

Some applications require a positive relationship between the prediction of the performer and each visual concept, *i.e.* each weight α_i must be a positive scalar. In this case, we use the cross-entropy between $\boldsymbol{\alpha}$ and $\mathbf{w}$ as the prior loss. In other cases, the MSE loss between $\boldsymbol{\alpha}$ and $\mathbf{w}$ is used as the loss. $\|\cdot\|_1$ and $\|\cdot\|_2$ denote the L-1 norm and L-2 norm, respectively.

In particular, in order to ensure $\alpha_i \geq 0$ in certain applications, we add a non-linear activation layer as the last layer of g, *i.e.* $\boldsymbol{\alpha} = \log[1 + \exp(x)]$, where x is the output of the last conv-layer.

3.1 Computation of prior weights $\mathbf{w}$

In this subsection, we will introduce two techniques to efficiently compute rough prior weights $\mathbf{w}$, which are oriented to the following two cases in application.

Case 1, filters in intermediate conv-layers of the performer are interpretable: As shown in Fig. 2(left), learning a neural network with interpretable filters is an emerging research direction in recent years. For example, (Zhang et al., 2018b) 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. 10 in the appendix for the visualization of filters). Thus, we can interpret the classification score of an object as a linear combination of elementary scores for the detection of object parts. Because such interpretable filters are automatically learned without part annotations, the quantitative explanation for the CNN (*i.e.* the performer) can be divided into the following two tasks: (i) annotating the name of the object part that is represented by each filter, and (ii) learning an explainer to disentangle the exact additive contribution of each filter (or each object part) to the performer output.

In this way, each f_i, $i = 1, 2, \ldots, n$, is given as an interpretable filter of the performer. According to (Zhang et al., 2018a), we can roughly represent the network prediction as

$$\hat{y} \approx \sum_i w_i y_i + b, \qquad \text{s.t.} \begin{cases} y_i &= \sum_{h,w} x_{hwi} \\ w_i &= \frac{1}{Z} \sum_{h,w} \frac{\partial \hat{y}}{\partial x_{hwi}} \end{cases} \tag{5}$$

where $x \in \mathbb{R}^{H \times W \times n}$ denotes a feature map of the interpretable conv-layer, and x_{hwi} is referred to as the activation unit in the location (h, w) of the i-th channel. y_i measures the confidence of detecting

the object part corresponding to the i-th filter. Here, we can roughly use the Jacobian of the network output *w.r.t.* the filter to approximate the weight w_i of the filter. Z is for normalization. Considering that the normalization operation in Equation (4) eliminates Z, we can directly use $\sum_{h,w} \frac{\partial \hat{y}}{\partial x_{hwi}}$ as prior weights $\mathbf{w}$ in Equation (4) without a need to compute the exact value of Z.

Case 2, neural networks for visual concepts share features in intermediate layers with the performer: As shown in Fig. 2(right), given a neural network for the detection of multiple visual concepts, using certain visual concepts to explain a new visual concept is a generic way to interpret network predictions with broad applicability. Let us take the detection of a certain visual concept as the target $\hat{y}$ and use other visual concepts as $\{y_i\}$ to explain $\hat{y}$. All visual concepts share features in intermediate layers.

Then, we estimate a rough numerical relationship between $\hat{y}$ and the score of each visual concept y_i. Let x be a middle-layer feature shared by both the target and the i-th visual concept. When we modify the feature x, we can represent the value change of y_i 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 push the feature towards the direction of boosting y_i, *i.e.* $\Delta x = \epsilon \frac{\partial y_i}{\partial x}$ (ϵ is a small constant), the change of the i-th visual concept 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 interpret different benchmark CNNs oriented to two different applications, in order to demonstrate the broad applicability of our method. In the first experiment, we used the detection of object parts to explain the detection of the entire object. In the second experiment, we used various face attributes to explain the prediction of another face attribute. We evaluated explanations obtained by our method qualitatively and quantitatively.

4.1 EXPERIMENT 1: USING OBJECT PARTS TO EXPLAIN OBJECT CLASSIFICATION

In this experiment, we used the method proposed in (Zhang et al., 2018b) to learn a CNN, where each filter in the top conv-layer represents a specific object part. We followed exact experimental settings in (Zhang et al., 2018b), which used the Pascal-Part dataset (Chen et al., 2014) to learn six CNNs for the six animal[2] categories in the dataset. Each CNN was learned to classify the target animal from random images. We considered each CNN as a performer and regarded its interpretable filters in the top conv-layer as visual concepts to interpret the classification score.

Four types of CNNs as performers: Following experimental settings in (Zhang et al., 2018b), we applied our method to four types of CNNs, including the AlexNet (Krizhevsky et al., 2012), the VGG-M, VGG-S, and VGG-16 networks (Simonyan & Zisserman, 2015), *i.e.* we learned CNNs for six categories based on each network structure. Note that as discussed in (Zhang et al., 2018b), skip connections in residual networks (He et al., 2016) 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 conv-layer, so we set $n = 256, 512, 512, 512$ for these networks. We used the masked output of the top conv-layer as x and plugged x to Equation (5) to compute $\{y_i\}$[1]. We used the 152-layer ResNet (He et al., 2016)[3] as g to estimate weights of visual concepts[4]. 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.

[2]Previous studies (Chen et al., 2014; Zhang et al., 2018b) usually selected animal categories to test part localization, because animals usually contain non-rigid parts, which present significant challenges for part localization.

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

[4]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.

	Experiment 1				Experiment 2				
	AlexNet	VGG-M	VGG-S	VGG-16	attractive	makeup	male	young	Avg.
Baseline	3.636	4.957	3.962	4.218	3.185	3.120	3.139	3.100	3.136
Ours	**5.199**	**5.908**	**5.913**	**6.054**	**3.216**	**3.134**	**3.139**	**3.160**	**3.162**

Table 1: Entropy of contribution distributions estimated by the explainer. A lower entropy of contribution distributions reflects more significant bias-interpreting. Our method suffered much less from the bias-interpreting problem than the baseline. Please see the appendix for more results.

		Experiment 1				Experiment 2				
		AlexNet	VGG-M	VGG-S	VGG-16	attractive	makeup	male	young	Avg.
Classification accuracy	Performer	93.9	94.2	95.5	95.4	81.5	92.3	98.7	88.3	90.2
	Explainer	93.6	94.0	94.9	96.6	76.0	88.8	97.6	82.8	87.1
Relative deviation	Performer	0	0	0	0	0	0	0	0	0
	Explainer	0.045	0.040	0.041	0.038	0.163	0.100	0.067	0.139	0.117

Table 2: Classification accuracy and relative deviations of the explainer and the performer. We used relative deviations and the decrease of the classification accuracy to measure the information that could not be explained by pre-defined visual concepts. Please see the appendix for more results.

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. Firstly, we evaluated the correctness of the estimated explanation for the performer prediction. In fact, there is no ground truth about exact reasons for each prediction. We showed example explanations of for a qualitative evaluation of explanations. We also used grad-CAM visualization (Selvaraju et al., 2017) of feature maps to prove the correctness of our explanations (see the appendix). In addition, we normalized the absolute contribution from each visual concept as a distribution of contributions $c_i = |\alpha_i y_i| / \sum_j |\alpha_j y_j|$. We used the entropy of contribution distribution $H(\mathbf{c})$ as an indirect evaluation metric for bias-interpreting. A biased explainer usually used very few visual concepts, instead of using most visual concepts, to approximate the performer, which led to a low entropy $H(\mathbf{c})$.

Secondly, we also measured the performer information that could not be represented by the visual concepts, which was unavoidable. We proposed two 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 $\sum_i \alpha_i y_i + b$. Another 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 $|\hat{y}_I - \sum_i \alpha_{I,i} y_{I,i} - b| / (\max_{I' \in \mathbf{I}} \hat{y}_{I'} - \min_{I' \in \mathbf{I}} \hat{y}_{I'})$, where $\hat{y}_{I'}$ denotes the performer's output for the image I'.

Considering the limited representation power of visual concepts, the relative deviation on an image reflected inference patterns, which were not modeled by the explainer. The average relative deviation over all images was reported to evaluate the overall representation power of visual concepts. Note that our objective was **not** to pursue an extremely low relative deviation, because the limit of the representation power is an objective existence.

4.2 Experiment 2: explaining face attributes based on face attributes

In this experiment, we learned a CNN based on the VGG-16 structure to estimate face attributes. We used the Large-scale CelebFaces Attributes (CelebA) dataset (Liu et al., 2015) to train a CNN 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 those representing global features of the face, *i.e. attractive*, *heavy makeup*, *male*, and *young*. It is because global features can usually 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 Experiment 1 (expect for $n = 39$) as g to estimate weights. We followed the Case-2 implementation in Section 3.1 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 $\beta = 0.2$ and used the L-2 norm loss in Equation (4) to learn all explainers. We used the same evaluation metric as in Experiment 1.

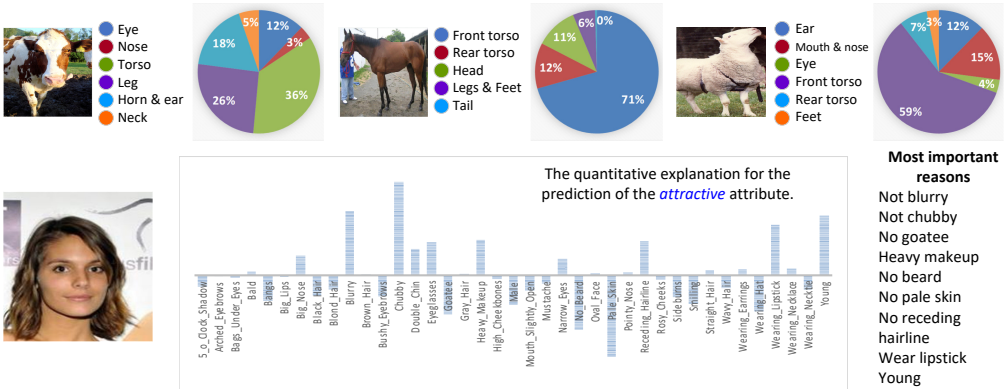

Figure 3: Quantitative explanations for the object classification (top) and the face-attribution prediction (bottom) made by performers. For performers oriented to object classification, we annotated the part that was represented by each interpretable filter in the performer, and we assigned contributions of filters $\alpha_i y_i$ to object parts (see the appendix). Thus, this figure illustrates contributions of different object parts. All object parts made positive contributions to the classification score. Note that in the bottom, 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, $\alpha_i y_i > 0$. Please see the appendix for more results.

4.3 Experimental results and analysis

We compared our method with the traditional baseline of only using the distillation loss to learn the explainer. Table 1 evaluates bias-interpreting of explainers that were learned using our method and the baseline. In addition, Table 2 uses the classification accuracy and relative deviations of the explainer to measure the representation capacity of visual concepts. Our method suffered much less from the bias-interpreting problem than the baseline.

Fig. 3 shows examples of quantitative explanations for the prediction made by the performer. We also used the grad-CAM visualization (Selvaraju et al., 2017) of feature maps of the performer to demonstrate the correctness of our explanations in Fig. 9 in the appendix. In particular, Fig. 4 in the appendix illustrates the distribution of contributions of visual concepts $\{c_i\}$ when we learned the explainer using different methods. Compared to our method, the distillation baseline usually used very few visual concepts for explanation and ignored most strongly activated interpretable filters, which could be considered as bias-interpreting.

5 Conclusion and discussions

In this paper, we focus on a new task, *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 compute quantitative 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 the 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 and produces similar prediction scores as the performer.

In particular, over-interpreting is the biggest challenge of using an additive explainer to interpret another neural network. In this study, we design two losses to overcome the bias-interpreting problems. Besides, in experiments, we also measure the amount of the performer knowledge that could not be represented by visual concepts in the explainer.

REFERENCES

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

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

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.

X. Chen, R. Mottaghi, X. Liu, S. Fidler, R. Urtasun, and A. Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts. *In CVPR*, 2014.

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.

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.

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

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

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

Tommaso Furlanello, Zachary C. Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born again neural networks. *In arXiv:1805.04770*, 2018.

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

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.

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

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.

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

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *In NIPS*, 2012.

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.

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

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

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

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

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

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

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

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.

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

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

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

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

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.

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

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.

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*, 2017a.

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

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

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

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

Q. Zhang, W. Wang, and S.-C. Zhu. Examining cnn representations with respect to dataset bias. *In AAAI*, 2018a.

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

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*, 2018c.

Quanshi Zhang, Yu Yang, Ying Nian Wu, and Song-Chun Zhu. Interpreting cnns via decision trees. *In arXiv:1802.00121*, 2018d.

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

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

DETAILED RESULTS

	AlexNet		VGG-M		VGG-S		VGG-16	
	Baseline	Ours	Baseline	Ours	Baseline	Ours	Baseline	Ours
bird	3.681	**5.201**	4.543	**5.885**	3.988	**5.870**	4.165	**6.040**
cat	3.294	**5.103**	5.604	**5.902**	3.954	**5.913**	4.894	**6.056**
cow	3.607	**5.236**	4.616	**5.941**	3.786	**5.901**	3.763	**6.084**
dog	3.774	**5.243**	5.510	**6.023**	4.087	**5.954**	4.570	**6.061**
horse	3.745	**5.278**	4.722	**5.921**	3.802	**5.973**	4.064	**6.060**
sheep	3.716	**5.135**	4.747	**5.773**	4.152	**5.869**	3.850	**6.022**
Avg.	3.636	**5.199**	4.957	**5.908**	3.962	**5.913**	4.218	**6.054**

Table 3: Entropy of contribution distributions. The entropy of contribution distributions reflects the level of bias-interpreting. The lower entropy indicates a larger bias. Our method suffered much less from the bias-interpreting problem than the baseline.

	AlexNet			VGG-M			VGG-S			VGG-16		
	Performer	Baseline	Ours	Performer	Baseline	Ours	Performer	Baseline	Ours	Performer	Baseline	Ours
bird	92.8	92.8	93.8	96.8	97.3	97.8	96.5	97.0	96.0	97.3	98.5	98.8
cat	96.3	95.7	95.2	94.3	95.7	95.5	95.3	95.5	96.0	94.3	97.0	97.0
cow	93.4	92.6	93.4	95.2	94.2	94.7	94.4	94.2	93.7	91.1	97.0	95.4
dog	92.5	92.5	92.0	93.8	94.7	94.5	95.3	95.5	93.0	94.5	95.0	94.5
horse	91.4	89.1	88.1	92.9	88.9	88.9	92.9	91.7	92.7	99.2	96.5	97.7
sheep	97.2	97.2	99.2	92.2	92.7	92.5	98.5	97.0	98.2	96.0	95.7	96.0
Average	93.9	93.3	**93.6**	94.2	93.9	**94.0**	95.5	**95.2**	94.9	95.4	96.6	**96.6**

	Attractive	Makeup	Male	Young	Avg.
Performer	81.5	92.3	98.7	88.3	90.2
Explainer, baseline	73.2	**89.0**	97.6	81.8	85.4
Explainer, ours	**76.0**	88.8	**97.6**	**82.8**	**86.3**

Table 4: Classification accuracy of the explainer and the performer. We use the the classification accuracy to measure the information loss when using an explainer to interpret the performer. Note that the additional loss for bias-interpreting successfully overcame the bias-interpreting problem, but did not decrease the classification accuracy of the explainer. Another interesting finding of this research is that sometimes, the explainer even outperformed the performer in classification. A similar phenomenon has been reported in (Furlanello et al., 2018). A possible explanation for this phenomenon is given as follows. When the student network in knowledge distillation had sufficient representation power, the student network might learn better representations than the teacher network, because the distillation process removed abnormal middle-layer features corresponding to irregular samples and maintained common features, so as to boost the robustness of the student network.

	AlexNet		VGG-M		VGG-S		VGG-16	
	Baseline	Ours	Baseline	Ours	Baseline	Ours	Baseline	Ours
bird	0.050	0.045	0.033	0.035	0.030	0.031	0.040	0.034
cat	0.041	0.040	0.029	0.030	0.036	0.039	0.027	0.030
cow	0.047	0.048	0.058	0.062	0.041	0.047	0.051	0.061
dog	0.041	0.047	0.025	0.026	0.041	0.047	0.028	0.026
horse	0.044	0.043	0.051	0.049	0.045	0.043	0.036	0.034
sheep	0.045	0.049	0.036	0.038	0.035	0.037	0.032	0.041
Average	0.045	0.045	0.039	0.040	0.038	0.041	0.036	0.038

Table 5: Relative deviations of the explainer. The additional loss for bias-interpreting successfully overcame the bias-interpreting problem and just increased a bit (ignorable) relative deviation of the explainer.

IMAGE-SPECIFIC EXPLANATIONS V.S. GENERIC EXPLANATIONS

(Zhang et al., 2018d) used a tree structure to summarize the inaccurate rationale of each CNN prediction into generic decision-making models for a number of samples. This method assumed the significance of a feature to be proportional to the Jacobian *w.r.t.* the feature, which is quite problematic. This assumption is acceptable for (Zhang et al., 2018d), because the objective of (Zhang et al.,

2018d) is to learn a generic explanation for a group of samples, and the inaccuracy in the explanation for each specific sample does not significantly affect the accuracy of the generic explanation. In comparisons, our method focuses on the quantitative explanation for each specific sample, so we design an additive model to obtain more convincing explanations.

VISUALIZATION OF BIAS-INTERPRETING

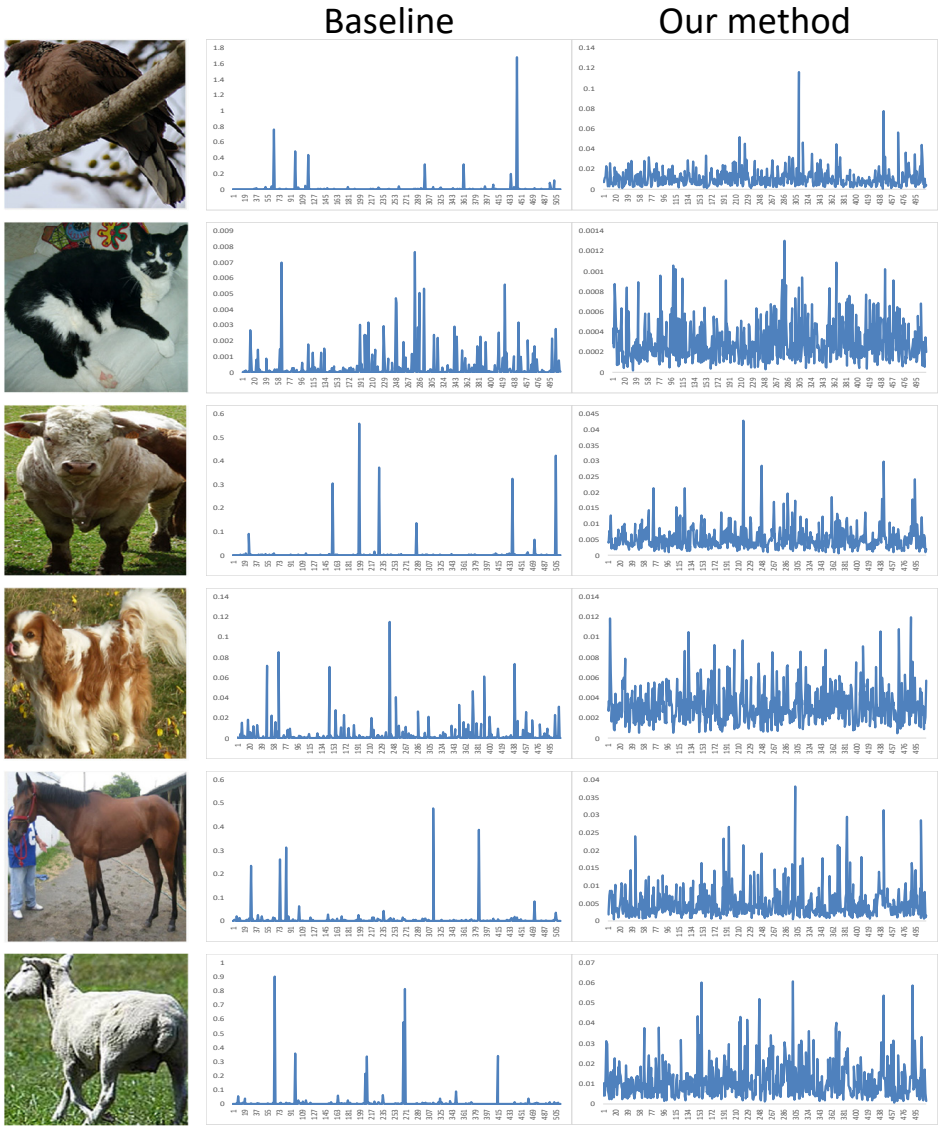

Figure 4: We compared the contribution distribution of different visual concepts (filters) that was estimated by our method and the distribution that was estimated by the baseline. The baseline usually used very few visual concepts to make predictions, which was a typical case of bias-interpreting. In comparisons, our method provided a much more reasonable contribution distribution of visual concepts.

VISUALIZATION OF QUANTITATIVE EXPLANATIONS

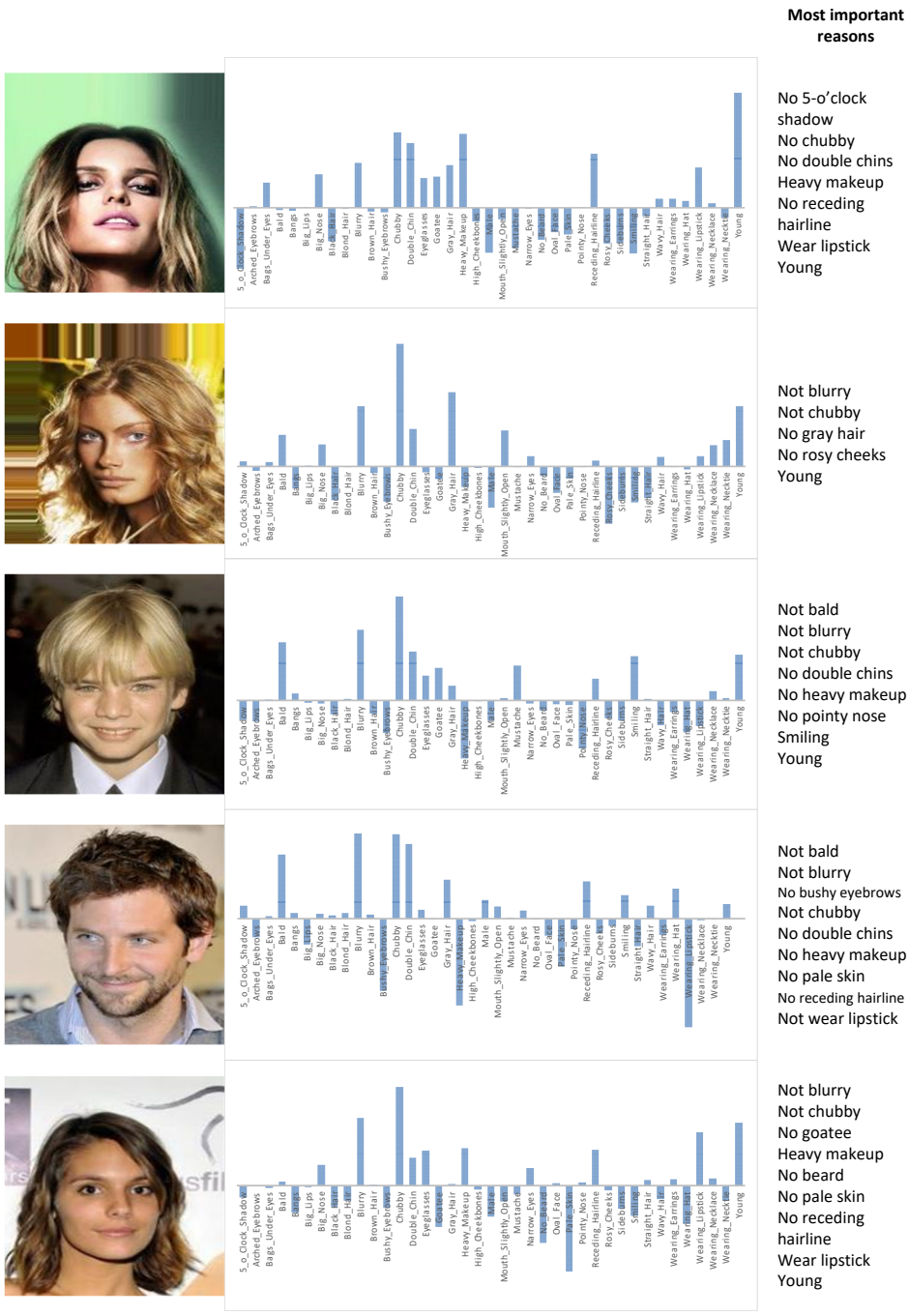

Figure 5: Quantitative explanations for the *attractive* attribute. Bars indicate elementary contributions $\alpha_i y_i$ from features of different face attributes, rather than the prediction of these attributes. For example, the network predicts a negative *goatee* attribute $y_{\text{goatee}} < 0$, and this information makes a positive contribution to the target *attractive* attribute, $\alpha_i y_i > 0$.

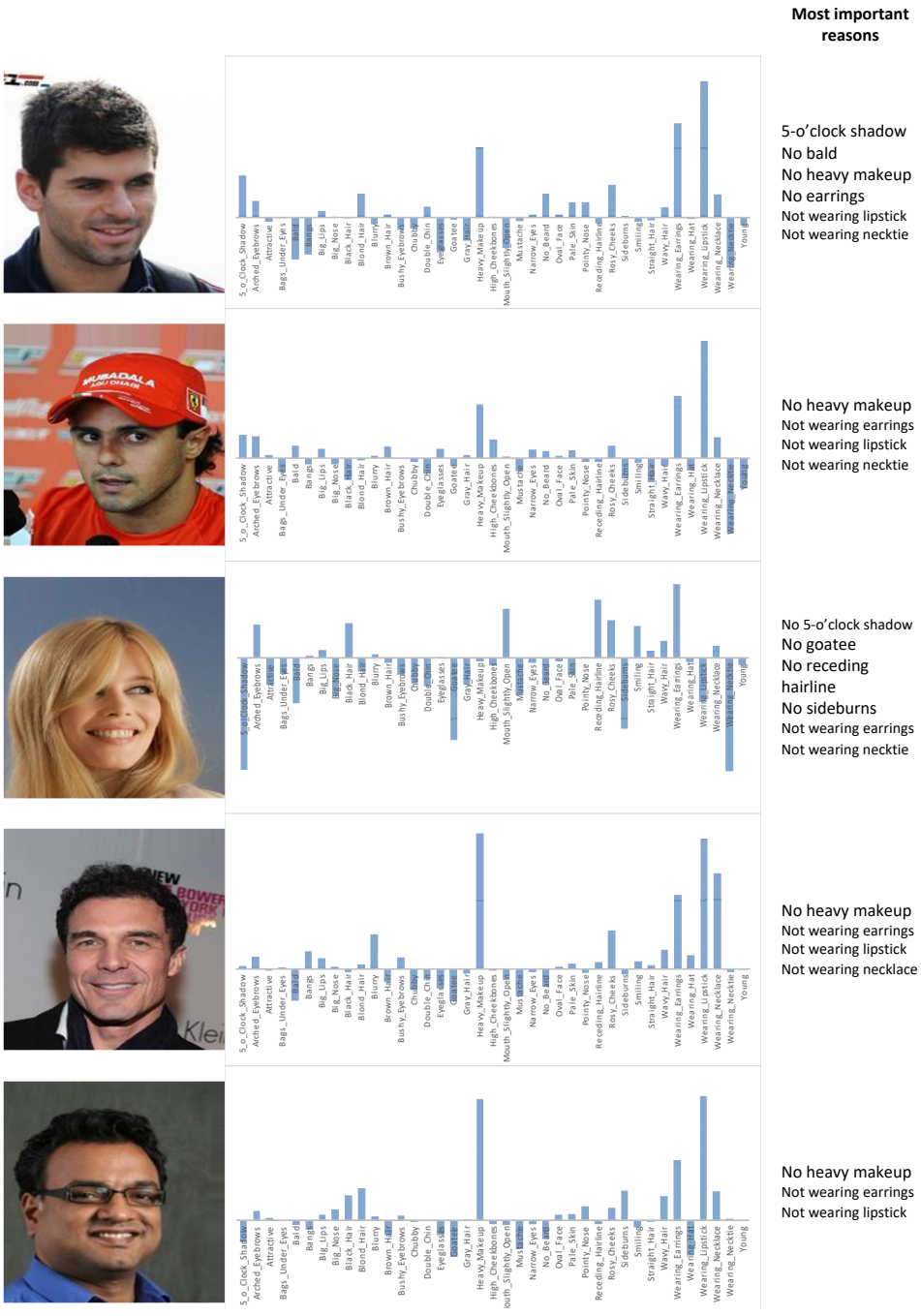

Figure 6: Quantitative explanations $\alpha_i y_i$ for the *male* attribute.

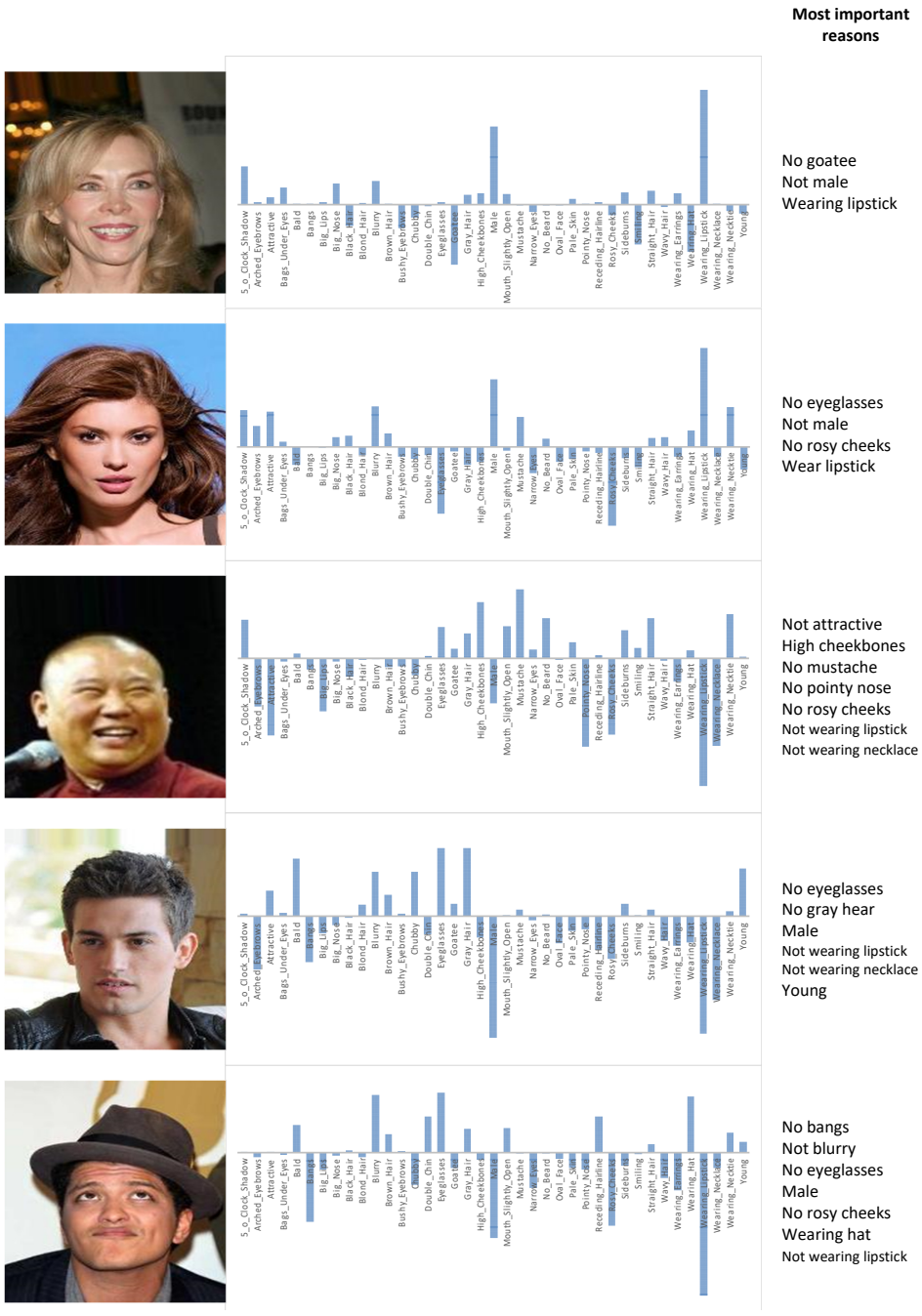

Figure 7: Quantitative explanations $\alpha_i y_i$ for the *heavy makeup* attribute.

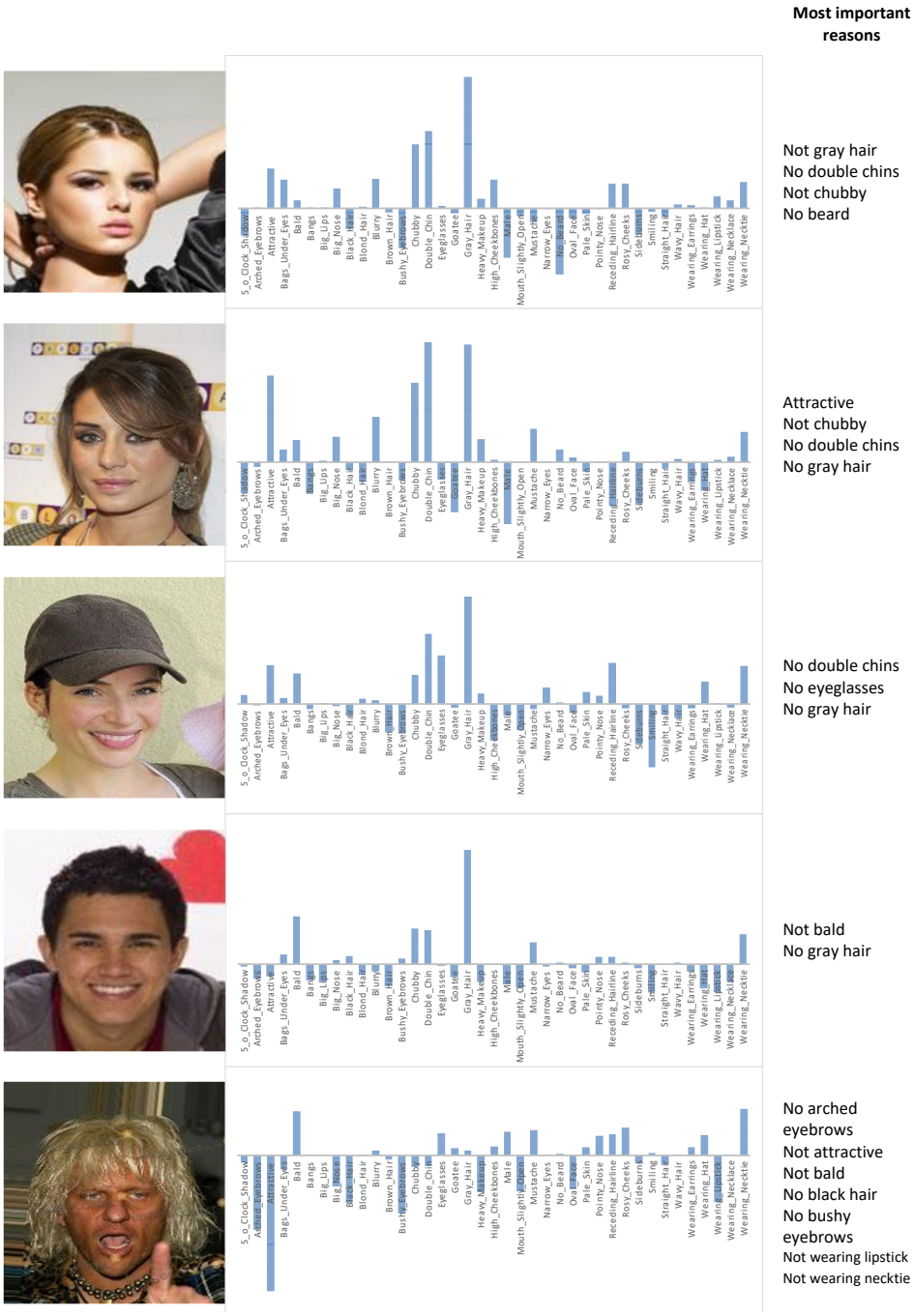

Figure 8: Quantitative explanations $\alpha_i y_i$ for the *young* attribute.

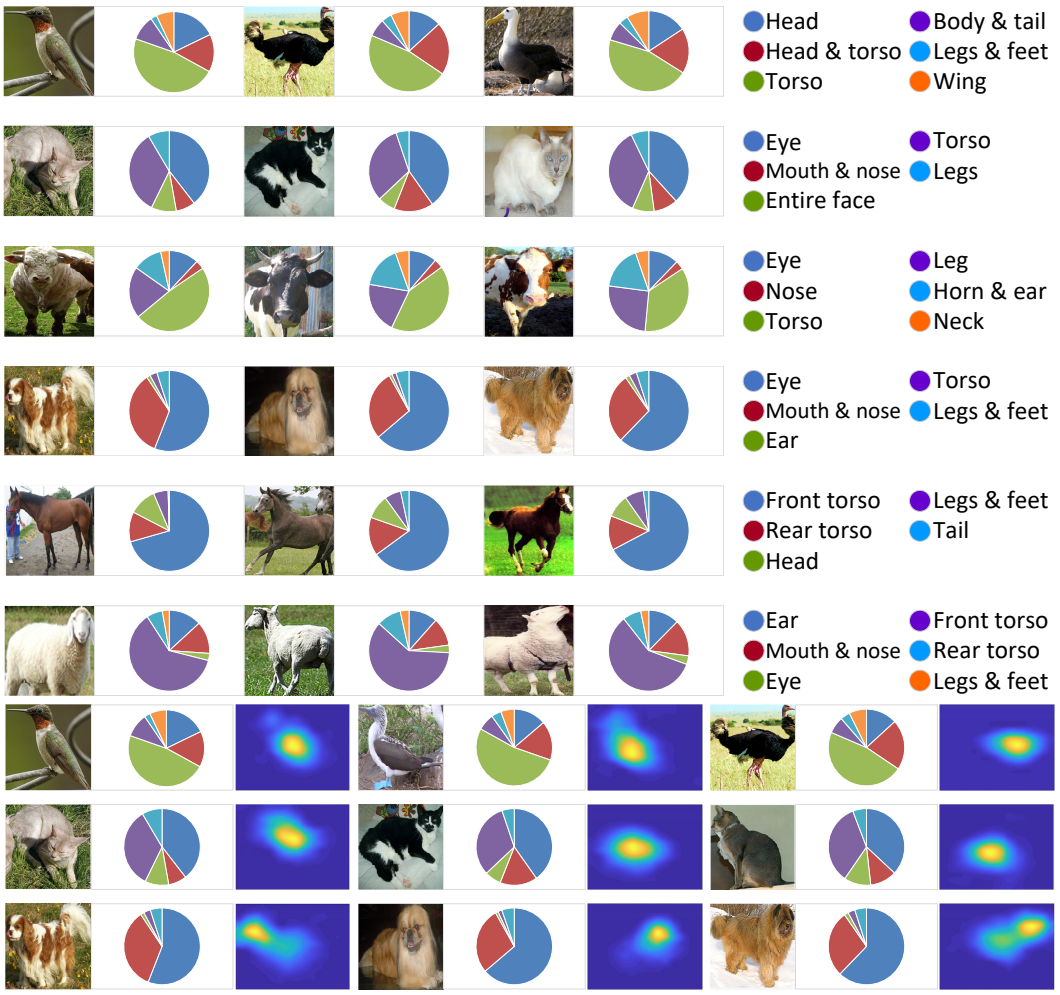

Figure 9: Quantitative explanations for object classification. We assigned contributions of filters to their corresponding object parts, so that we obtained contributions of different object parts. According to top figures, we found that different images had similar explanations, *i.e.* the CNN used similar object parts to classify objects. Therefore, we showed the grad-CAM visualization of feature maps (Selvaraju et al., 2017) on the bottom, which proved this finding.

VISUALIZATION OF INTERPRETABLE FILTERS

Figure 10: We visualized interpretable filters in the top conv-layer of a CNN, which were learned based on (Zhang et al., 2018b). 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.

ANNOTATIONS OF PART NAMES OF INTERPRETABLE FILTERS

(Zhang et al., 2018b) learned a CNN, where each filter in the top conv-layer represented a specific object part. Thus, we annotated the name of the object part that corresponded to each filter based on visualization results (see Fig. 10 for examples). We simply annotate 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.

Then, we assigned the contribution of a filter to its corresponding part, *i.e.* $Contri_{part} = \sum_{i\text{:i-th filter represents the part}} \alpha_i y_i$.

DETAILS IN EXPERIMENT 1

We changed the order of the ReLU layer and the mask layer after the top conv-layer, *i.e.* placing the mask layer between the ReLU layer and the top conv-layer. According to (Zhang et al., 2018b), this operation did not affect the performance of the pre-trained performer. We used the output of the mask layer as x and plugged x to Equation (5) to compute $\{y_i\}$.

Because the distillation process did not use any ground-truth class labels, the explainer's output $\sum_i \alpha_i y_i + b$ was not sophisticatedly learned for classification. Thus, we used a threshold $\sum_i \alpha_i y_i + b > \tau$ ($\tau \approx 0$), instead of 0, as the decision boundary for classification. τ was selected as the one that maximized the accuracy. Such experimental settings made a fairer comparison between the performer and the explainer.