Accurate Parts Visualization for Explaining CNN Reasoning via Semantic Segmentation

Ren Harada
r-harada@mi.t.u-tokyo.ac.jp
Antonio Tejero-de-Pablos
antonio-t@mi.t.u-tokyo.ac.jp
Tatsuya Harada
harada@mi.t.u-tokyo.ac.jp

Graduate School of Information Science and Technology, The University of Tokyo
Tokyo, Japan

Abstract

Nowadays, neural networks are often used for image classification, but the structure of their decisions is difficult to understand because of their "black-box" nature. Different visualization techniques have been proposed to provide additional information on the reason of the classification results. Existing methods provide quantitative explanations by calculating heatmaps and interpretable components in the image. While the latter provides semantics on the image parts that contribute for the classification, the component areas are blurry due to the use of linear layers, which do not consider surrounding information. This makes hard to point out the specific reason for the classification and to evaluate quantitatively. In this paper, we introduce a novel method for explaining classification in neural networks, the Parts Detection Module. Unlike previous methods, ours is capable of determining the accurate position of the interpretable components in the image by performing upsampling and convolution stepwise, similarly to semantic segmentation. In addition to providing quantitative visual explanations, we also proposed a method to verify the validity of the quantitative explanations themselves. The experimental results prove the effectivity of our explanations.

1 Introduction

In recent years, Convolutional Neural Networks (CNNs) have been successfully applied to a variety of image classification tasks [6]. However, while CNNs achieve highly accurate predictions, it is not straightforward to understand the process and reason for the produced classifications by directly observing the weights of CNN. This is due to the discrepancy between how humans and CNNs recognize objects; while CNNs employ the RGB information of the image pixels, humans resort to the semantics contained in the parts of the objects. Applications of CNNs in autonomous driving and healthcare would largely benefit from such explainability, as a misclassification could lead to serious accidents. Determining the reason of a faulty classification would allow understanding the situation and determining effective measures against it. Therefore, the cues used by CNNs when taking a decision should be presented and explained in a form that is linked to semantic information rather than a list of observable numbers.

Grad-CAM [11] has been traditionally used to provide explanations of CNN decisions. This method allows visualizing the area of the input image that the network considers to be important for classification. Since an image area does not provide enough cues for a complete understanding, Interpretable Basis Decomposition (IBD) [17] decomposes the area into smaller areas with semantic meaning, called interpretable basis, which correspond to each of the components of the class the image belongs to. For example, a *living room* is classified as such since it contains a *coffee table*, a *couch*, a *TV stand*, etc. IBD reconstructs the output of Grad-CAM with the linear summation of its sub-areas, where the weights of the linear summation can be regarded as the contribution rate of each sub-area, making it possible to evaluate quantitatively the importance of each component. However, since IBD uses a linear layer to determine the components in the features extracted by the CNN, it loses the spatial information. This causes the sub-areas to be located in a position different from the actual position of the corresponding component when it is visualized. This also prevents us from validating correctly the contribution rates indicated quantitatively. Furthermore, instead of using vague pixel areas, providing the exact region of the semantic component is desirable for a clear explanation.

In this paper, we propose a novel method for explaining CNN classifications, the Semantic Parts Explainer (SPE), which detects the specific parts that the CNN relies on for classification via semantic segmentation. Our Dictionary Learning module reconstructs a Grad-CAM-like output by using the detected parts directly as an interpretable basis (Fig. 1). Our method creates different bases with respect to the input image, considering the spatial information modeled in the convolutions by using a convolutional layer for the detecting parts (instead of a linear layer), and quantifies the contribution rate of each part. The contributions are summarized as follows. 1) We propose the Semantic Parts Detector, a novel method to generate visual explanations create a basis considering the spatial information of the CNN features. 2) Leveraging the spatial precision of our semantic parts, we propose a novel evaluation method for quantitative explanations. 3) Finally, we prove the validity of our visual explanations and the contribution rate of their parts by evaluating it via the method in 2).

2 Related Works

There are mainly two ways of achieving comprehensibility in deep neural networks: through indirect understanding and through direct understanding.

1. Indirect understanding of the decision structure of a network involves replacing the reasoning process with an understandable model, for example, a decision tree [10] that reproduces the network we want to understand (i.e., a CNN) [2, 15].

2. Direct understanding of the decision structure of a network involves controlling the inputs and outputs to understand the reasoning of the network, without changing its inner structure. [1, 3, 11, 12]

2.1 Indirect Understanding

Methods for indirect understanding [2, 15] first gather the outputs of the network we want to explain, and then use these outputs as supervised data to train a decision tree to perform a

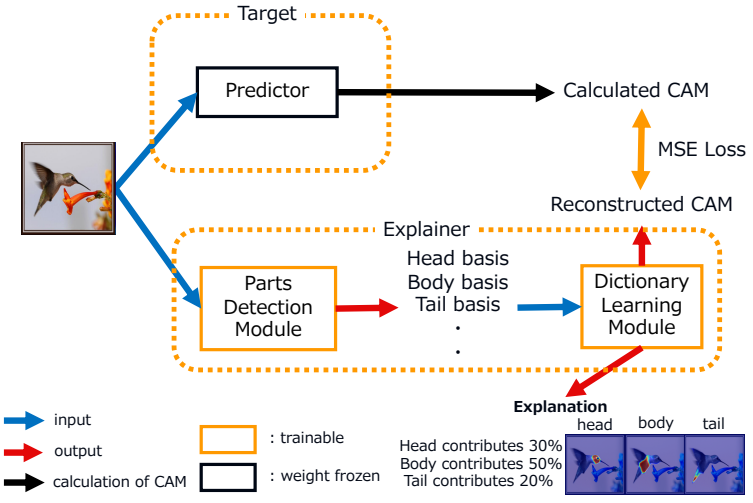

Figure 1: Overview of our method. It consists of two main parts: the Predictor, which is the network whose reasoning we are trying to explain, and the Explainer, which generates the explanation. The Predictor outputs the results for the image classification problem (e.g., bird classification) and calculates their respective Class Activation Maps (CAMs). The Explainer receives the CAM from the Predictor, and generates the component bases corresponding to the semantic parts in the input image (e.g., head and wings in the case of birds) via a Parts Detection Module, and reconstructs the CAM by the linear sum of these bases. Then, after reconstructed, we calculate the contribution of each part and use it to generate the explanation.

similar reasoning (i.e., to output similar values). Decision trees are used since they learn an architecture that is easily explainable.

The method in [15] transforms the CNN into a network capable of understanding the reasoning process by changing the inner structure. The model becomes understandable by learning each channel of the intermediate features obtained from the input image in order to identify parts of the image with a semantic meaning. Taking a dog image as an example, the intermediate features of the middle layer identifies the dog's ears, another channel identifies the head, and another output identifies the tail. This way, when there is a misclassification, we can explain which image part was the cause by checking which channel was activated. This allows for an indirect understanding since we know which parts were identified by the middle layer.

These indirect methods aim at understanding the decision structure of the network, but they do not fully reproduce the original network. Therefore, if the network we want to explain and the network prepared as an alternative are very different, we may have a wrong understanding of the network decision mechanism

2.2 Direct Understanding

Methods for direct understanding [9, 11, 13, 17] involve using an attention mechanism that allows visualizing the image areas that a network considers important when making decisions. This is achieved by conveying gradients of a specific class to the target convolutional

layers of the network we try to explain.

Another method creates two versions of the network we want to explain [1], one from the input to a shallow/intermediate layer and the other from the input to a deep/final layer, and classifies the image by adding a linear layer at the end. This allows us to understand the classes that each layer is able to classify.

Direct methods overcome the problems of indirect methods as they do not change the structure of the original network.

Our proposed method belongs to the group of direct understanding.

Visualization Explanations

Visualization explanations, such as Grad-CAM [11], have become a mainstream explanation method for direct understanding, since they combine a simple architecture and an intuitive comprehensibility. In [16], a gradient approach to explaining importance through visualization was proposed, by using the backpropagation of gradients and the properties of the global average pooling layer (GAP layer). The GAP layer is applied to the CNN to generate a Class Activation Map (CAM) that represents the relevance of each image pixel for the classification.

IBD[17] was also proposed to provide explanations through visualization. IBD decomposes the CAM computed using Grad-CAM into an interpretable basis of several concepts, that is, a vector linked to semantic information such as colors and part names. This mechanism allows explaining which concepts the CNN considered important via the contribution rate of how much the decomposed basis affects the reconstruction of the CAM. To create the basis of concepts, IBD performed multi-label classification for each pixel by using a linear layer. However, this causes spatial information to be lost, and thus, the output activation maps become vague area of the image. This hinders the quality of the explanations, as parts are not properly represented.

In this paper, we leverage semantic segmentation to visualize the exact position of the concepts, or parts. For the creation of the basis, instead of a linear layer, we employ a convolutional layer like in semantic segmentation models, which allows considering the surrounding pixels (i.e., modeling spatial information). This allows for a more precise representation of the image parts in the explanations.

3 Method

We propose a method for explaining the reasoning of a CNN by visualizing the exact image parts considered for classification, the Semantic Parts Detector (SPD). Unlike previous methods, which offer a vague CAM area, SPD achieves a representation of the actual position of the image parts by modeling spatial information for creating the basis of components. We quantitatively verify the validity of the contribution rate of each part to the classification.

In order to detect the accurate image parts that contain a semantic concept for classification, we pretrain our Parts Detection module via a dataset annotated with the segmented parts. The Predictor provides the classification results for the input image, which are used to calculate the CAM via Grad-CAM [11]. Then, the Parts Detection module detects the image parts to construct a basis of components. The CAM from the acquired bases is reconstructed using the Dictionary Learning module. The contribution rate of each part is calculated from the coefficients of the lexical learning module used in reconstructing the CAM. The resulting

explanation of the classification consists of the aforementioned contribution rate along with the accurate visualization of their corresponding parts in the image.

3.1 Parts Detection module

This module generates a set of basis from the input image, by detecting the image parts with a semantic meaning. For our implementation, instead of using simple linear transformations, we use ENCNet[14]. ENCNet explicitly learns information about which parts tend to exist at the same time for a given class, by using a Context Encoding module. For example, the parts "beak" and "wings" are typical of the class "bird", and the parts "hands" and "hair" are typical of the class "human".

Compared with the state-of-the-art explanation method IBD [17], there is a significant difference in the upsampling process (Fig. 2). IBD applies a linear layer and a single final upsample to output their basis, which is similar to the segmentation result. On the other hand, ENCNet upsamples and convolutes the intermediate features multiple times, thus, retaining more spatial features than IBD. This aims to provide clearer and more accurate parts as our bases.

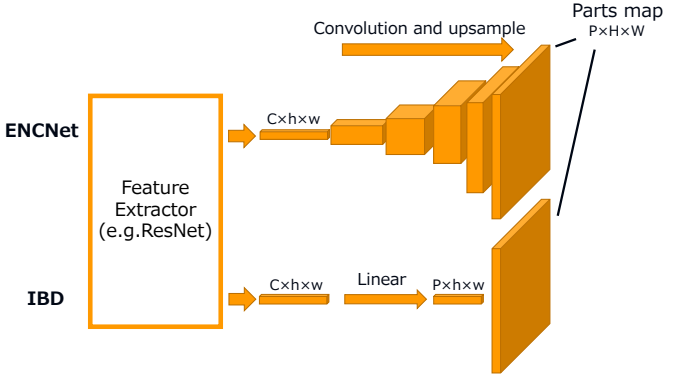

Figure 2: Details of our Parts Detection module (upper) vs. the state of the art (IBD). First, both architectures extract intermediate features ($C \times h \times w$) from the input image using a CNN (e.g. ResNet). ENCNet upsamples and convolutes the intermediate features multiple times to obtain an accurate map of the image parts. In contrast, IBD applies a linear layer and a single upsampling, which hinders the modeling of spatial features and lowers the quality of the detected parts.

3.2 Dictionary Learning Module

The Dictionary Learning module reconstructs a CAM using the bases acquired by the Parts Detection module. It is inspired by dictionary learning [9], a learning method in which the target signal is reconstructed by a linear sum of a group of signals called a dictionary. In this case, the acquired basis is used as a dictionary and the CAM calculated by Grad-CAM is reconstructed. Assuming that the basis corresponding to part p_i is $M_{p_i} \in R^{HW}$ and the weight

of the linear summation is q_{p_i}, the reconstructed CAM M_{rec} is

$$M_{rec} = \sum_i q_{p_i} M_{p_i} \tag{1}$$

Using q_{p_i} (learned during training), the contribution rate u_{p_i} of the part p_i is

$$u_{p_i} = \frac{q_{p_i}}{\sum_i q_{p_i}} \tag{2}$$

We found that the contribution rate of the background was often high when we use standard MSE loss. In order to solved it, we used the initialization of weights as follows,

$$q^0_{p_i} = min\left(\frac{max_{i,j}(M_{p_i})}{k}, 1\right) \tag{3}$$

where M_{p_i} is the output of the Parts Detection module (i.e., a mask with shape batch_size x H x W), and k is a hyperparameter. The purpose of this initialization is to fix the contribution rates of the parts that are not present in the image to zero, Then the hyperparameter k is introduced to control the order of initial weights.

4 Experiments

4.1 Datasets

We pretrained our Parts Detection module using the Pascal Part dataset [4]. Each image in the dataset is labeled with a single class, out of the 20 different image classes contained. Additionally, each image contains a number between 1 and 15 part classes. Similarly, each pixel is labeled with a part class. We selected 16 different parts for our experiments (see Sec. 4.2). This sums up to a total of 4638 images, from which 10% were used for testing.

The Predictor (network we want to explain) was trained using the Caltech-UCSDBirds200 (CUB200) dataset. The dataset contains 200 classes of bird images, adding up to 4758 images for training and 6033 images for testing. Thus, we employ two datasets in our paper, as in the most relevant work to ours[7, 15, 17].

4.2 Settings

We chose ResNet101 [6] as the target architecture for explaining its classification decisions (Predictor). ResNet101 has achieved an accuracy of 0.758 for 20 classes in the Pascal Part dataset, and 0.745 in the CUB200 dataset. For our Parts Detection module we compared the performance of ENCNet with a baseline, which is used in IBD. We chose the 16 most frequent part classes from the Pascal Part dataset: *eye*, *nose*, *leg*, *mouth*, *face*, *neck*, *arm*, *hand*, *torso*, *hair*, *beak*, *wing*, *plant*, *tail* and *background*. Parts shared among image classes were assigned the same label (e.g., dog/human legs or human/bird eyes). Both ENCNet and the baseline(IBD) were trained using three GPUs, Quadro RTXs with a batch size of 64 and an image size of (480×480). We used the AdaBound[8] optimizer for 100 epochs, with an initial learning rate of 0.001 and a final learning rate of 0.2. We used cross entropy loss as loss function.

The Dictionary Learning module was trained using the RMSProp [5] optimizer for 50 epochs per image to obtain the contribution rate of each part, with a learning rate of 0.01 and a hyperparameter k of 10. We used mean squared loss as loss function.

4.3 Evaluation

We evaluated our method quantitatively using the Average Drop Percent (ADP) [3] and applied it to evaluate the ranking of parts. This metric assigns a higher score to visualization maps that correctly highlight the pixels that are relevant for determining the class c. If the $(i,j) \in R^{H \times W}$ pixel of CAM M_c computed for the class c by a visualization method (e.g., Grad-CAM) is below a certain threshold θ, then we replace the same i,j pixel of the input image I with zero, and input it to the Predictor CNN to obtain the classification results O_c. Let Y_c be the classification results for the original I, ADP is expressed as:

$$\text{ADP} = \frac{\max(0, Y_c - O_c)}{Y_c} \times 100 \tag{4}$$

In order to perform a quantitative evaluation of the parts detected by our Parts Detection module, we sorted them by their contribution rates, and calculated a CAM M_p for the top $p \in P$ parts, using $\theta = 0.3$. Let $O_{c,p}$ be the classification results when the pixels of M_p below the θ become zero in I. In this case, ADP is

$$\text{ADP} = \sum_{p}^{P_{top\ n}} \frac{\max(0, Y_c - O_{c,p})}{Y_c} \times 100 \tag{5}$$

We calculate an ADP_{rand} by using an O obtained by *randomly* replacing with zero the same number of pixels as in ADP. We define ADP_{diff} as the difference between both:

$$\text{ADP}_{diff} = \text{ADP}_{rand} - \text{ADP} \tag{6}$$

For each test image N, we compute

$$\text{ADP}_{mean} = \frac{1}{N} \sum_{i=1}^{N} \text{ADP}_{diff}^{(i)} \tag{7}$$

If $\text{ADP}_{mean} > 0$, we determine that the important features are visualized.

4.4 Results

First, we evaluated quantitatively our Parts Detector module. Our evaluation metric is the mean IoU, that is, we calculate the intersection area of the output mask (segmentations from the Parts Detector) and the ground-truth mask (annotations from Pascal Parts) and divide it by the union area. Then we calculate the average for all classes:

$$\text{meanIoU} = \frac{1}{N_C} \sum_{c}^{C} \frac{intersection\ of\ class\ masks}{union\ of\ class\ masks} \tag{8}$$

The mean IoU achieved using the test set of Pascal Parts is 0.758 for ENCNet and 0.320 for the baseline(IBD). ENCNet significantly outperformed baseline(IBD) because this method can model better the parts that belong to a specific class.

Next, we evaluate quantitatively the explanations generated for the Pascal Part dataset and CUB200. The results of the ADP_{mean} described in Sec. 4.3 are shown in Table1. We ranked our bases (detected parts) by taking those with the highest contribution rate, and calculated the corresponding APDs for a $P = 5$ and $P = 10$. The average ADP in the top 10

Table 1: ADP quantifies the decrease in classification accuracy by comparing adding noise to random parts of the input image with adding noise to the visualized important parts of the image. APD assumes that the classification accuracy decreases when important image parts are lost. A high value of ADP_{mean} indicates that the calculated bases allow for a successful visualization of the important sections of the image for classification.

	ADP	ADP random	ADP mean
Pascal Part Dataset top 5	85.5	89.6	4.09
Pascal Part Dataset top 10	84.2	87.0	2.87
CUB200 top 5	110.3	121.7	11.3
CUB200 top 10	111.7	117.5	5.82

bases of the Pascal Part dataset is about 2.87 higher than the randomly added ADP, and the ADP in the top 5 bases is 4.09 higher. The difference between the ADP and the ADP random of the top 5 bases is larger than that of the top 10 bases, suggesting that the top 5 bases retain more important features. Similarly, the average ADP of the top 10 bases in CUB200 is 5.82 higher than that of the randomly substituted ADP, and the ADP of the top 5 bases is 11.3 higher. Following the same reasoning as with the Pascal Part dataset, the top 5 bases prove to retain more important features.

Next, we use the CUB200 dataset to qualitatively evaluate the visualization results of the parts with the top contribution rate (Fig. 3). Each image represents an interpretable basis and its inferred part label. The top 5 parts are sorted from left to right in decreasing order of contribution rate. The upper row shows that our method can successfully detect the accurate parts that better identify the input image. The lower row shows the results of the state-of-the-art baseline IBD [17], whose detected parts are vague and blurry, especially beak and neck. Since the detected bases do not properly segment the image areas for beak and neck, IBD determines a larger contribution rate on these parts, even though they are not actually relevant for classification. For this reason, our explanations are more accurate, outperforming the state of the art.

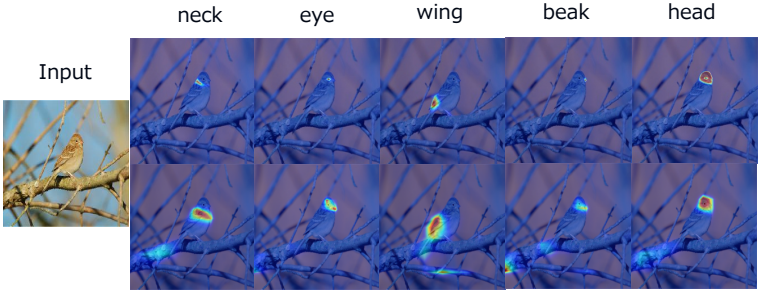

Figure 3: Comparison of the bases detected by our SPD (upper) and IBD [17] (lower): The convolutional architecture of our Parts Detector achieves more accurate bases, in particular eye and beak, whereas the bases of the comparison method are vague and blurry.

Fig. 4 shows a qualitative evaluation of our generated explanations, which consist of parts contribution rates (semantics) and reconstructed CAMs (visualization). From left to right,

the input image and its respective CAM and reconstructed cam are shown. Below, there are the three parts with the highest contribution rate. The contribution for *beak* is higher in pelicans due to their long beaks (lower left), and the contribution for *head* and *eyes* is higher in birds with distinctive facial patterns (upper). In some cases, unrelated parts such as *tree* have high contribution rates, since CAM pays attention to image areas that are not relevant (lower right).

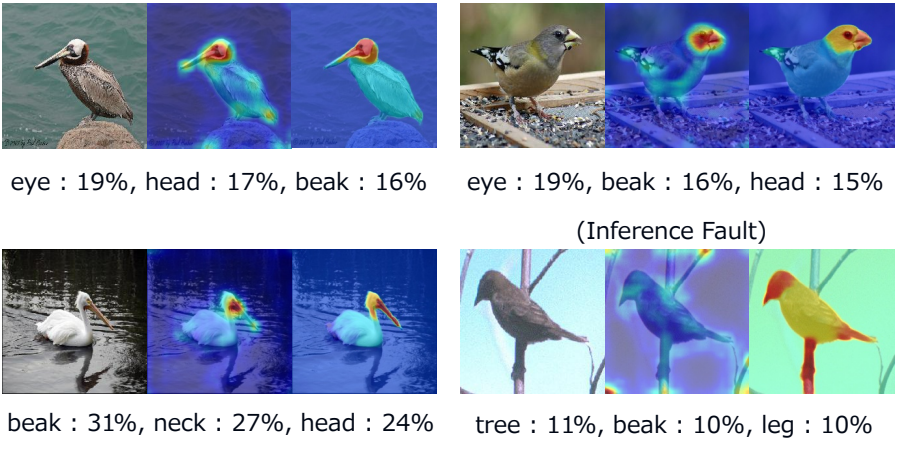

Figure 4: Qualitative comparison of our visual explanations: input image (left), computed CAM (middle), and reconstructed CAM (right). The detected parts with the top 3 contributions appear below the image.

5 Discussion and Conclusion

We evaluated quantitatively our Parts Detection module to show its effectiveness in matching the position of the visualized parts with the position of the actual parts in the image. This is because its convolutional architecture allows considering spatial features. We also evaluated quantitatively our Dictionary Learning module to show its effectiveness in reconstructing the CAM visualization through the image areas containing the concept bases with the highest contribution rates. This is because the module succeeds in learning how to give more importance to the parts with higher weights.

In this paper, we proposed a novel method to accurately visualize which image parts are important for classification at the pixel level, and also to specify their contribution at the part level. This allows creating an explanation on the reasoning of the prediction network judgment, by connecting the semantics of the image parts with the classification results of the CNN we want to explain. More concretely:

- By using a convolutional layer (the ENCNet segmentation network in our case) instead of a linear layer, and our weight initialization, we are able to generate a basis that takes spatial information into account.
- The use of spatial information-conscious bases enables quantitative evaluation using methods such as ADP.

To summarize, our proposed method enables the generation of a positionally accurate bases and the quantitative evaluation of the validity of the contribution rate of each basis, by applying a semantic segmentation method that uses convolutional layers instead of linear layers for the generation of bases. The quantitative evaluation showed that the generated contributions were in the correct order and that the basis visualizes the important parts of the image.

As our future work, we consider providing detailed explanations, such as color and shape of the parts, unsupervisedly.

6 Acknowledgements

This work was partially supported by JST AIP Acceleration Research Grant Number JPMJCR20U3, and partially supported by JSPS KAKENHI Grant Number JP19H01115. We thank four reviewers for their professional reviews and their time. We also thank Yu Ono and Keita Otani for their helpful comments on the implementation of our method.

References

[1] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3319–3327, July 2017. doi: 10.1109/CVPR.2017.354.

[2] Leo Breiman, Nong Shang, Tommaso Furlanello, and Lei Jimmy Ba. Born again trees. 1996.

[3] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 839–847, March 2018. doi: 10.1109/WACV.2018.00097.

[4] 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. 06 2014. doi: 10.1109/CVPR.2014.254.

[5] Alex Graves. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850, 2013. URL http://arxiv.org/abs/1308.0850.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition*, 2016. doi: 10.1109/CVPR.2016.90.

[7] Lisa Hendricks, Ronghang Hu, Trevor Darrell, and Zeynep Akata. Grounding visual explanations, 07 2018.

[8] Liangchen Luo, Yuanhao Xiong, Yan Liu, and Xu Sun. Adaptive gradient methods with dynamic bound of learning rate. *CoRR*, abs/1902.09843, 2019. URL http://arxiv.org/abs/1902.09843.

[9] Julien Mairal, Jean Ponce, Guillermo Sapiro, Andrew Zisserman, and Francis R. Bach. Supervised dictionary learning. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1033–1040. Curran Associates, Inc., 2009. URL http://papers.nips.cc/paper/3448-supervised-dictionary-learning.pdf.

[10] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[11] 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 *IEEE International Conference on Computer Vision*, 2017. doi: 10.1007/s11263-019-01228-7.

[12] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin A. Riedmiller. Striving for simplicity: The all convolutional net. *CoRR*, abs/1412.6806, 2014.

[13] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. pages 3156–3164, 2014.

[14] H. Zhang, K. Dana, J. Shi, Z. Zhang, X. Wang, A. Tyagi, and A. Agrawal. Context encoding for semantic segmentation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7151–7160, June 2018. doi: 10.1109/CVPR.2018.00747.

[15] Quanshi Zhang, Yu Yang, Haotian Ma, and Ying Nian Wu. Interpreting cnns via decision trees, 2018.

[16] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2921–2929, June 2016. doi: 10.1109/CVPR.2016.319.

[17] Bolei Zhou, Yiyou Sun, David Bau, and Antonio Torralba. Interpretable basis decomposition for visual explanation. In *European Conference on Computer Vision*, 2018.